ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ
Σχολή Θετικών Επιστημών και Τεχνολογίας
Τμήμα Επιστήμης και Τεχνολογίας Υπολογιστών

Μεταπτυχιακή Διπλωματική Εργασία

Μελέτη και Υλοποίηση Πλατφόρμας Ασαφοποίησης Οντολογιών

ΠΑΠΑΦΡΑΓΚΟΣ ΙΩΑΝΝΗΣ

ΑΜ:2009024

Επιβλέποντες Καθηγητές: **κ. Βασιλάκης Κωνσταντίνος**
κ. Γουάλλες Εμμανουήλ

Τρίπολη, Μάιος 2011

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ
Σχολή Θετικών Επιστημών και Τεχνολογίας
Τμήμα Επιστήμης και Τεχνολογίας Υπολογιστών

Μεταπτυχιακή Διπλωματική Εργασία

Μελέτη και Υλοποίηση Πλατφόρμας Ασαφοποίησης Οντολογιών

ΠΑΠΑΦΡΑΓΚΟΣ ΙΩΑΝΝΗΣ

ΑΜ:2009024

Επιβλέποντες Καθηγητές: **κ. Βασιλάκης Κωνσταντίνος**
κ. Γουάλλες Εμμανουήλ

Τρίπολη, Μάιος 2011

Περίληψη

Στην παρούσα διπλωματική εργασία, πραγματοποιείται μια ποιοτική περιγραφή και ανάλυση του όρου Σημασιολογικού Ιστού και των τεχνολογιών αναπαράστασης γνώσης που υπόκεινται σε αυτόν, όπως δομή και χαρακτηριστικά οντολογιών, γλώσσες σημασιολογικής περιγραφής καθώς επίσης και μεθοδολογίες περιγραφής οντολογιών. Μεγάλη έμφαση δίνεται στην ασαφή γνώση που αποδίδεται μέσω της πληροφορίας που αναπαριστάται από μία οντολογία, λόγω της δυσκολίας που παρουσιάζεται, στην περιγραφή μιας πληροφορίας με μαθηματική ακρίβεια. Παρουσιάζονται όλα τα απαραίτητα στοιχεία για την κατανόηση του επιστημονικού όρου «Ασαφή Σύνολα» και της συσχετίσεως του με την ύπαρξη των οντολογιών.

Στο πλαίσιο της εκπόνησης της διπλωματικής εργασίας, υλοποιήθηκε μία διαδικτυακή πλατφόρμα ασαφοποίησης οντολογιών με την ονομασία «wbGraphFuzzyOnto», βασισμένη κυρίως σε τεχνολογίες μιας νέας εξελισσόμενης τάσης του διαδικτύου, το WEB 2.0. Κύριος στόχος της πλατφόρμας *wb*GraphFuzzyOnto είναι η καλύτερη αξιοποίηση των οντολογιών, αντιμετωπίζοντας το πρόβλημα της «Ασάφειας», έτσι ώστε να επιτύχουμε καλύτερα αποτελέσματα κατά την διαδικασία ανάλυσης και επεξεργασίας των πληροφοριών αυτών, σε επίπεδο μηχανής από τους υπολογιστές.

Λέξεις Κλειδία

Οντολογίες, Γλώσσες Σημασιολογικής Περιγραφής, f-OWL, Ασαφή Σύνολα, Μεθοδολογίες Περιγραφής Οντολογιών , Μεθοδολογία Ασαφοποίησης Οντολογιών IKARUS-Onto, Ajax Framework - ZK, Java J2EE, SVG, Jena Framework, DOT.

ABSTRACT

In the present postgraduate diploma thesis, a qualitative description and analysis is realised regarding the concepts of Semantic Web and for technologies of represantion of knowledge that define it, such as the structure and characteristics of ontologies, semantic description languages as well as methodologies for describing ontologies. Great emphasis is given to vague knowledge provided by the information represented by an ontology, because of the difficulty in describing information as an element of an ontology with mathematical precision. This paper presents all the necessary information for understanding the scientific concept of “Fuzzy Sets” and its association with the existence of ontology.

In the scope of this diploma thesis, a web-based platform was developed for developing fuzzy ontologies with the name "wbGraphFuzzyOnto", based mainly on technologies in a new evolutionary tendency in the space of the web, Web 2.0. The main objective of the "wbGraphFuzzyOnto" platform is the best use of ontologies tackling the problem of "Fuzzyness", so that we will be able to achieve better results in the process of analyzing and processing information, at the computer machine level.

Key Words

Ontologies, Semantic Description Languages, f-OWL, Fuzzy Sets, Methodologies for Ontology Development, Methodology IKARUS-Onto, Ajax Framework - ZK, Java J2EE, SVG, Jena Framework, DOT.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω του καθηγητές μου κ. Βασιλάκη Κωνσταντίνο και κ. Γουάλλες Εμμανουήλ, για την εμπιστοσύνη που μου έδειξαν με την ανάθεση της παρούσας διπλωματικής εργασίας, αλλά και την καθοδήγηση τους στην επιτυχή διεκπεραίωση της. Θα ήθελα ακόμη να ευχαριστήσω τον κ. Αλεξόπουλο Παναγιώτη (Knowledge Engineer and Researcher) για τις πολύτιμες ιδέες του, στην υλοποίηση της crisp οντολογίας.

Τέλος θα ήθελα να ευχαριστήσω τους γονείς μου για την ηθική και όχι μόνο συμπαράσταση που μου προσέφεραν καθ' όλη τη διάρκεια των μεταπτυχιακών μου σπουδών.

Περιεχόμενα

Ευρετήριο Πινάκων

Ευρετήριο Εικόνων

1 ΚΕΦΑΛΑΙΟ – ΕΙΣΑΓΩΓΗ

1.1 ΣΗΜΑΣΙΟΛΟΓΙΚΟΣ ΙΣΤΟΣ ΚΑΙ Η ΑΝΑΓΚΑΙΟΤΗΤΑ ΤΟΥ

Στον παγκόσμιο ιστό υπάρχουν εκατοντάδες εκατομμύρια ιστοχώροι, είναι διασυνδεδεμένοι δεκάδες εκατομμύρια υπολογιστές και καθημερινά το επισκέπτονται εκατοντάδες εκατομμύρια άνθρωποι από όλο τον κόσμο. Για αυτό τον λόγο έχει καταστεί η μεγαλύτερη πηγή πληροφορίας στις μέρες μας.

Μία όμως από τις βασικές αδυναμίες του παγκόσμιου ιστού είναι ότι η πληροφορία που είναι διαθέσιμη, δεν είναι κατανοητή από τους υπολογιστές, αφού τα δεδομένα είναι σχεδιασμένα για παρουσίαση και κατανόηση τους μόνο από τον άνθρωπο. Αυτό έχει σαν αποτέλεσμα να χαρακτηρίζεται ο παγκόσμιος ιστός σαν μία πηγή ασύνδετων πληροφοριών μεταξύ τους και σε συνδυασμό με την πληθώρα πληροφοριών που διακινούνται καθημερινά, να καθιστά επιτακτική την ανάγκη ανάλυση και κατανόηση τους από του υπολογιστές.

Χαρακτηριστικό παράδειγμα του παραπάνω προβλήματος είναι οι μηχανές αναζήτησης. Οι μηχανές αναζήτησης εφαρμόζουν αναζήτηση χρησιμοποιώντας κατά κανόνα μια πληροφορία λέξη – κλειδί, η οποία όμως είναι κατανοητή μόνο από τον άνθρωπο. Αυτό έχει σαν αποτέλεσμα να μην επιστρέφονται σχεδόν ποτέ τα επιθυμητά αποτελέσματα και να χαρακτηρίζονται τα αποτελέσματα αυτά από τον μεγάλο όγκο πληροφοριών, αμφίβολης πολλές φόρες σχέσης με το αντικείμενο αναζήτησης.

Την ανάγκη για την επίλυση του προβλήματος αυτού, στα πλαίσια και της ευρύτερης προσπάθειας που καταβάλλεται για την εξέλιξη του παγκόσμιου ιστού, έρχεται να επιλύσει ο διεπιστημονικός τομέας με την ονομασία «Σημασιολογικός Ιστός (Semantic Web)». Ο Σημασιολογικός Ιστός δεν αποτελεί μία ξεχωριστή οντότητα σε σχέση με τον σημερινό παγκόσμιο ιστό, αλλά αποτελεί μία προσπάθεια επέκτασης του. Σκοπός του είναι ο εμπλουτισμός και η απόδοση του περιεχομένου του παγκόσμιου ιστού, με σημασιολογικούς όρους (συγκεκριμενοποιώντας τους πόρους του διαδικτύου με την προσθήκη σημασιολογικών μεταδεδομένων), ακολουθώντας έναν δομημένο τρόπο έτσι ώστε να το καταστήσουν επεξεργάσιμο και κατανοητό σε επίπεδο μηχανής. Αυτό ακριβώς ήτανε ένα και από τα οράματα του εμπνευστή του παγκοσμίου ιστού (Tim Berners Lee - 1999) το να είναι σε θέση οι «μηχανές» να αναλύσουν και να επεξεργάζονται τα δεδομένα του ιστού.

«Ο Σημασιολογικός Ιστός δεν είναι ένας ξεχωριστός ιστός, αλλά μία επέκταση του υπάρχοντος, όπου ένα σαφές νόημα δίνεται στις πληροφορίες και που, ακόμα, επιτρέπει στους ανθρώπους να συνεργάζονται με τους υπολογιστές» [Tim Berners-Lee, James Hendler, Ora Lassila, the Semantic Web, Scientific American, May 2001].

Μεγάλη προσπάθεια επίσης καταβάλλεται με το να εκφραστεί η πληροφορία σημασιολογικά με τέτοιον τρόπο ώστε να ενισχυθεί η επαναχρησιμοποίηση της και ο διαμοιρασμός της. Επιτυγχάνοντας έτσι την ομαλή λειτουργία και συνδεσιμότητα ετερογενών συστημάτων και εφαρμογών, με απώτερο στόχο την καλύτερη υποστήριξη των ανθρώπων στην διεκπεραίωση ενεργειών.

Ο Σημασιολογικός Ιστός βασίζεται στις Οντολογίες. Μια Οντολογία απαρτίζεται από ένα σύνολο δεδομένων με σημασιακό περιεχόμενο. Μια Οντολογία αποτελεί ένα τρόπο μοντελοποίησης εννοιών και σχέσεων με διττό στόχο. Από την μία να εξασφαλίσει ένα κοινό λεξιλόγιο και από την άλλη να εξασφαλίσει μια κοινή κατανόηση του τομέα γνώσης τον οποίο μοντελοποιεί. Μία Οντολογία ουσιαστικά περιλαμβάνει κλάσεις, που είναι οτιδήποτε μπορεί να ειπωθεί κάτι για αυτό, στιγμιότυπα που αναπαριστούν στοιχεία και μπορεί να ανήκουν σε μία κλάση, σχέσεις που περιγράφουν αληθείς προτάσεις μεταξύ στιγμιότυπων αλλά και άλλων στοιχείων (λεκτικά).

Ένα από τα προβλήματα που συναντάμε κατά την δημιουργία μιας οντολογίας έχει να κάνει με την δυσκολία στο να περιγράψουμε με μαθηματική ακρίβεια τον κόσμο που μας περιβάλει. Την λύση σε αυτό το πρόβλημα μας την προσφέρουν τα «Ασαφή Σύνολα». Τα Ασαφή Σύνολα θεωρούνται ως ένα από τους πλέον εύστοχους τρόπους περιγραφής του υλικού και κοινωνικού μας συνόλου σε γλώσσα κατανοητή από του υπολογιστές.

Αξίζει να αναφερθούμε και σε μία άλλη προσπάθεια που γίνεται για την βελτίωση και την εξέλιξη του παγκόσμιου ιστού και η προσπάθεια αυτή εκφράζεται με τον όρο Web 2.0. Ο όρος Web 2.0 χρησιμοποιείται για να περιγράψει τη νέα γενιά του Παγκόσμιου Ιστού η οποία βασίζεται στην όλο και μεγαλύτερη δυνατότητα των χρηστών του Διαδικτύου να μοιράζονται πληροφορίες και να συνεργάζονται online[1]. Ουσιαστικά πρόκειται για μία δυναμική διαδικτυακή πλατφόρμα στην οποία μπορούν να αλληλεπιδρούν χρήστες χωρίς εξειδικευμένες γνώσεις σε θέματα υπολογιστών και δικτύων. Παρακάτω αναφέρουμε τις κυριότερες εφαρμογές του Web 2.0:

- Blogs (Ιστολόγια)
- Wikis
- Mashups
- SOA (Web Services)
- RSS feeds
- Social networking
- Tagging ("folksonomy")
- Peer-to-peer networking

Ίσως το πιο διαδεδομένο σύνολο τεχνολογιών ανάπτυξης εφαρμογών Web 2.0 στην παρούσα φάση είναι το *AJAX* (Asynchronous JavaScript and XML), το οποίο παρέχει τη δυνατότητα οι όποιες αλλαγές να γίνονται μόνο σε μεμονωμένα συστατικά της διεπαφής

του χρήστη (user interface) και όχι μέσω της πλήρους ανανέωσης της ιστοσελίδας στην οποία βρίσκεται ο χρήστης[2]. Αυτό δίνει το πλεονέκτημα, όχι μόνο μίας αναβαθμισμένης εμπειρίας αλληλεπίδρασης του χρήστη με το σύστημα λόγω του ότι η επικοινωνία δεν διακόπτεται από το φόρτωμα νέων ιστοσελίδων, αλλά και σημαντικής μείωσης στην συσσωρευμένη κίνηση του δικτυού. Ένα από τα βασικά πλεονεκτήματα της τεχνολογίας AJAX είναι ότι μπορεί να λειτουργήσει ανεξαρτήτως πλατφόρμας και κατά συνέπεια έχει την δυνατότητα να λειτουργεί στην πλειοψηφία των συστημάτων που έχουν οι χρήστες[3].

1.2 Σκοπος ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Σκοπός της παρούσας διπλωματικής εργασίας είναι η περιγραφή και η ανάδειξη των πλεονεκτημάτων χρήσης τεχνολογιών που βασίζονται στον Σημασιολογικό Ιστό αλλά κυρίως η δημιουργία μιας διαδικτυακής πλατφόρμας με την ονομασία «*wb*GraphFuzzyOnto» βασισμένη κυρίως σε τεχνολογίες Web 2.0. Κύριος στόχος της πλατφόρμας *wb*GraphFuzzyOnto είναι η καλύτερη αξιοποίηση των οντολογιών, αντιμετωπίζοντας το πρόβλημα της «Ασάφειας» δηλαδή της δυσκολίας που συναντάμε στην διατύπωση με μαθηματική ακρίβεια των πληροφοριών που απαρτίζουν μια οντολογία έτσι ώστε να επιτύχουμε καλύτερα αποτελέσματα, κατά την διαδικασία ανάλυσης και επεξεργασίας των πληροφοριών αυτών σε επίπεδο μηχανής από τους υπολογιστές.

1.3 Οργανωση Κειμενου

Η παρούσα διπλωματική εργασία οργανώνεται με τον ακόλουθο τρόπο:

- Το κεφάλαιο 2, αναφέρεται στο θεωρητικό υπόβαθρό που είναι απαραίτητο για την ανάπτυξη του θέματος, της παρούσας διπλωματικής εργασίας.
- Το κεφάλαιο 3, παρουσιάζει τις βασικότερες μεθοδολογίες περιγραφής συμβατικών οντολογιών.
- Το κεφάλαιο 4, αναλύει και περιγράφει την προτεινόμενη μεθοδολογία ανάπτυξης ασαφών οντολογιών.
- Το κεφάλαιο 5, πραγματοποιείται μία ανάλυση των απαιτήσεων του συστήματος, καθορισμός λειτουργιών ανά ομάδα χρηστών που απευθύνονται και τέλος επισήμανση των βασικότερων λειτουργικών περιορισμών.
- Το κεφάλαιο 6, περιγράφει τις τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση της πλατφόρμας «wbGraphFuzzyOnto» και τα κριτήρια επιλογής τους.
- Στο κεφάλαιο 7, περιγράφονται διεξοδικά όλες οι μονάδες που απαρτίζουν την πλατφόρμα «wbGraphFuzzyOnto».
- Στο κεφάλαιο 8, πραγματοποιείται μία προσομοίωση της πλατφόρμας «wbGraphFuzzyOnto», βάση ενός συγκεκριμένου σεναρίου χρήσης.

- Το κεφάλαιο 9, συνοψίζει την εργασία, εκθέτοντας τα συμπεράσματα που αποκομίσαμε καθώς και πιθανές μελλοντικές επεκτάσεις.
- Το κεφάλαιο 10, παρουσιάζει το σύνολο της βιβλιογραφίας και των δικτυακών τόπων, που χρησιμοποιήθηκαν ως πηγές για την εκπόνηση της παρούσας διπλωματικής εργασίας.
- Στο κεφάλαιο 11, παρουσιάζεται η διαδικασία εγκατάσταση της πλατφόρμας «wbGraphFuzzyOnto».

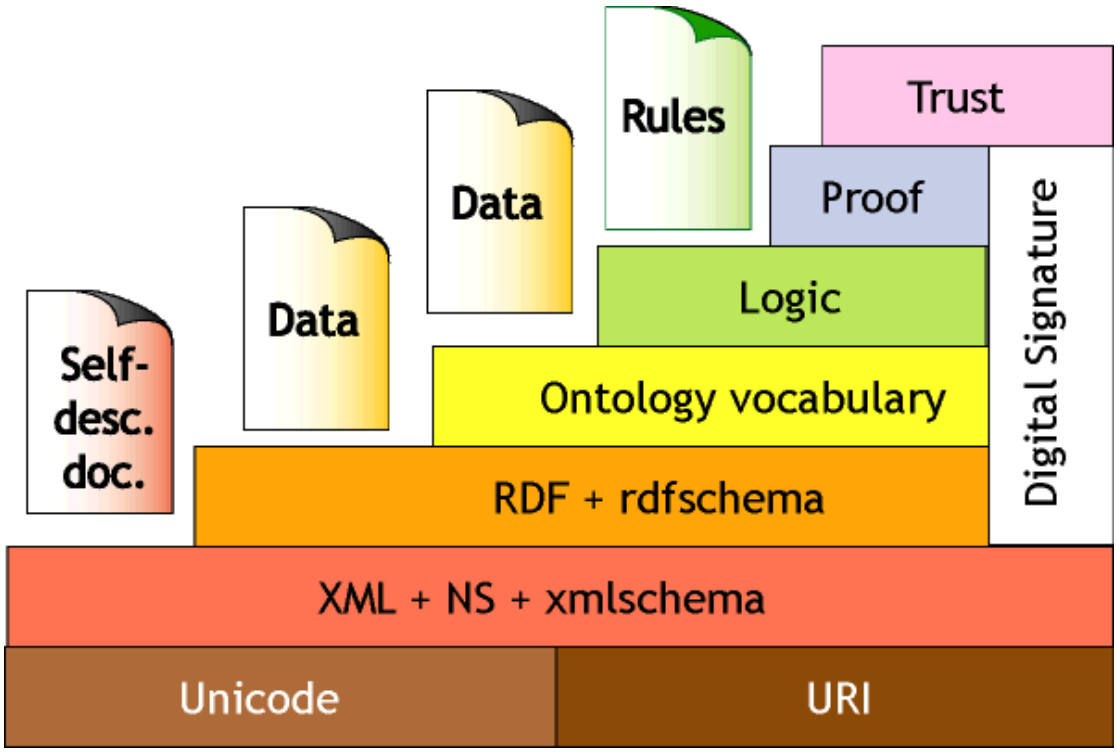
2 Θεωρητικο Υποβαθρο

2.1 Αρχιτεκτονικη Σημασιολογικου Ιστου

Όπως αναφέραμε στο κεφάλαιο 1 ο Σημασιολογικός Ιστός αποτελεί μια προσπάθεια μετεξέλιξης και επέκτασης του παγκόσμιου ιστού (World Wide Web). Η προσπάθεια αυτή αποσκοπεί στην οργάνωση και ταξινόμηση των πληροφοριών που είναι διασκορπισμένες στο διαδίκτυο, με ένα τρόπο καθορισμένο από αναγνωρισμένα διεθνή πρότυπα έτσι ώστε οι πληροφορίες να είναι κατανοητές και επεξεργάσιμες σε επίπεδο μηχανών. Οι μηχανές θα είναι σε θέση να μην επεξεργάζονται τα δεδομένα πλέον μηχανικά αλλά θα μπορούν να τα κατανοούν ταυτοποιώντας με συγκεκριμένη σημασία και περιεχόμενο.

Ο Σημασιολογικός Ιστός αναμένεται να συμβάλει στην καθιέρωση μιας νέας γενιάς μηχανών αναζήτησης. Η συγκεκριμένη κατηγορία μηχανών αναζήτησης δεν θα βασίζεται αποκλειστικά και μόνο σε μία πληροφορία λέξη – κλειδί άλλα θα λαμβάνεται υπόψη και η σημασιολογία της συγκεκριμένης πληροφορίας. Αυτό θα έχει σαν αποτέλεσμα μία πιο ορθολογική αναζήτηση αλλά και τον περιορισμό του μεγάλου όγκου των αποτελεσμάτων εξαλείφοντας πληροφορίες αμφίβολης σχέσης με το αντικείμενο αναζήτησης.

Στην εικόνα 2.1 παρουσιάζεται η αρχιτεκτονική του Σημασιολογικού Ιστού όπως αυτή αποδόθηκε από τον Tim Berners-Lee. Βασικό χαρακτηριστικό κάθε επιπέδου που απαρτίζουν την αρχιτεκτονική του Σημασιολογικού Ιστού είναι η άμεσης εξάρτηση του από το αμέσως κατώτερο επίπεδο στο οποίο βασίζεται και το οποίο επεκτείνει.



Εικόνα 2.1 – Αρχιτεκτονική Σημασιολογικού Ιστού

Συγκεκριμένα:

- **1^{ο} Επίπεδο**

Ο Σημασιολογικός Ιστός θεμελιώνεται πάνω στην ήδη υπάρχουσα υποδομή του Ιστού: στο πρωτόκολλο HTTP για τη μεταφορά, στα URIs (Universal Resource Indicators, Καθολικό Αναγνωριστικό όλων των διαδικτυακών Πόρων) για την ονοματολογία, στην κωδικοποίηση Unicode (Universal Code) για καθολική προσπέλαση.

- **2^{ο} Επίπεδο**

Το συγκεκριμένο επίπεδο σχετίζεται με την γλώσσα XML. Πρόκειται για μία ευρέως χρησιμοποιούμενη περιγραφική γλώσσα για την αναπαράσταση δεδομένων στο διαδίκτυο.

- **3^{ο} Επίπεδο**

Σε αυτό το επίπεδο συναντάμε δύο γλώσσες αναπαράσταση δεδομένων, την RDF και RDFS, που αποτελούν μια επέκταση της γλώσσας XML. Με την RDF μπορεί κάποιος να αναπαραστήσει ισχυρισμούς της μορφής υποκείμενο – ιδιότητα – αντικείμενο , ενώ η RDFS αποτελεί ουσιαστικά την «γραμματική» για την σύνταξη της RDF.

- **4^{ο} Επίπεδο**

Μια κοινή αναπαράσταση για τις *οντολογίες* (ontologies), που επιτρέπουν στους όρους που χρησιμοποιούνται στο επίπεδο δεδομένων (3^{ο} Επίπεδο) να ορίζονται και να συσχετίζονται μεταξύ τους (RDFS, DAML+OIL, OWL).

- **5^{ο} Επίπεδο**

Είναι το επίπεδο της λογικής που επιτρέπει την λογική συσχέτιση μεταξύ διαφορετικών διαδικτυακών πόρων.

- **6^{ο} Επίπεδο**

Το συγκεκριμένο επίπεδο είναι το επίπεδο των αποδείξεων που επιτρέπει την εξαγωγή συμπερασμάτων χρησιμοποιώντας τα υπάρχοντα δεδομένα.

- **7^{ο} Επίπεδο**

Τέλος, το επίπεδο της *εμπιστοσύνης* (trust), όπου σε συνδυασμό με την τεχνολογία των *ψηφιακών υπογραφών* (digital signatures), θα εξασφαλίζει το βαθμό στον οποίο οι πληροφορίες που διακινούνται, επεξεργάζονται και συμπεραίνονται στο Σημαντικό Ιστό είναι αξιόπιστες, με αυτοματοποιημένο τρόπο.

2.2 ΟΝΤΟΛΟΓΙΕΣ

Ο Σημασιολογικός Ιστός, όπως έχουμε ήδη αναφέρει βασίζεται στις οντολογίες. Τον όρο «Οντολογία» τον συναντάμε τόσο στον κλάδο της φιλοσοφίας όσο και στον κλάδο της πληροφορικής. Έχουν δοθεί πολλοί ορισμοί σχετικά με τον όρο της «Οντολογίας» από τους επιστήμονες, ίσως ο πιο διαδεδομένος και σαφής ορός είναι αυτός που αποδόθηκε από τον Thoma R. Gruber[13]: «Οντολογία είναι ένας τυπικός και σαφής ορισμός μιας κοινά αποδεκτής εννοιολογικής μορφοποίησης». Με τον όρο εννοιολογική μορφοποίηση αναφερόμαστε στο γνωστικό αντικείμενο ή στην γνωστική περιοχή την οποία θέλουμε να περιγράψουμε.

Ο ορισμός που έχει αποδοθεί και υιοθετηθεί από τα μέλη που αποτελούν το W3C είναι ο ακόλουθος[14]: Μια οντολογία είναι μια αυστηρά μαθηματική περιγραφή ενός συγκεκριμένου πεδίου γνώσης και περιλαμβάνει ένα σύνολο από όρους και συσχετίσεις μεταξύ τους. Η οντολογία που σχεδιάζουμε για ένα πεδίο ενδιαφέροντος (domain), χρησιμοποιείται τόσο για να υπάρχει ένα κοινά αποδεκτό λεξιλόγιο πεδίου όσο και για να μπορέσουμε να εξάγουμε συμπεράσματα μεταξύ των στοιχείων που απαρτίζουν την οντολογία εκμεταλλευόμενοι την σημασιολογία του κάθε στοιχείου. Πιο συγκεκριμένα, μια οντολογία παρέχει περιγραφές για:

- Ιδιότητες που περιγράφουν τις κλάσεις και συνήθως τις συσχετίζουν μεταξύ τους.
- Στιγμιότυπα των κλάσεων αυτών.
- Κλάσεις (έννοιες) που αφορούν το πεδίο γνώσης.

2.2.1 *ΠΛΕΟΝΕΚΤΗΜΑΤΑ ΧΡΗΣΗΣ ΟΝΤΟΛΟΓΙΩΝ*

Τα πλεονεκτήματα που προκύπτουν από την χρήση και την ανάπτυξη των οντολογιών είναι πολύ σημαντικά και είναι τα ακόλουθα[15]:

- *Κοινή χρήση της δομής της πληροφορίας ανάμεσα σε ανθρώπους και μηχανές.* Είναι ουσιαστικά ο σημαντικότερος λόγος για την ανάπτυξη οντολογιών. Η δυνατότητα των υπολογιστών να καταλαβαίνουν την πληροφορία που περιγραφεί μια οντολογία, έχει σαν αποτέλεσμα των συνδυασμό πληροφοριών ως απάντηση στις αναζητήσεις των χρηστών.
- *Επαναχρησιμοποίηση της γνώσης μίας γνωστικής περιοχής.* Όταν μία οντολογία είναι καλά ορισμένη και διατυπωμένη, τότε δίνεται η ευχέρεια σε διαφορές ομάδες χρηστών να την επαναχρησιμοποιήσουν είτε για να την επεκτείνουν είτε για συνένωση με ήδη υπάρχουσες οντολογίες.
- *Σαφής ορισμός των εννοιών μιας γνωστικής περιοχής.* Ο σαφής ορισμός των εννοιών μιας γνωστικής περιοχής βοηθά τους νέους χρήστες να κατανοήσουν πολύ πιο εύκολα την πληροφορία που περιγράφεται και τους καθίστα ικανούς στο να προβούν στην επεξεργασία της πληροφορίας αυτής.

- *Διαχωρισμός της γνώσης του κάθε πεδίου από τη λειτουργική γνώση.* Μπορούμε να απομονώσουμε την διαδικασία της λειτουργίας μιας υπηρεσίας ή ενός αντικειμένου και να την περιγράψουμε σαν μία ξεχωριστή οντολογία και στην συνέχεια να την χρησιμοποιήσουμε αυτούσια ως βάση για την περιγραφή άλλων υπηρεσιών ή αντικειμένων που ακολουθούν την ίδια διαδικασία στην λειτουργία τους.

- *Ανάλυση της γνώσης της κάθε περιοχής.* Είναι μία διαδικασία η οποία μπορεί να πραγματοποιηθεί με *σχετική ευκολία, εφόσον είναι διαθέσιμες και σαφείς οι προδιαγραφές για την ορολογία που χρησιμοποιούνται.*

2.2.2 *ΔΟΜΗ ΟΝΤΟΛΟΓΙΑΣ*

Τα βασικά στοιχεία που χαρακτηρίζουν μια οντολογία και με τα οποία μπορούμε να ορίσουμε μία οντολογία είναι τα ακόλουθα:

- *Κλάσεις(classes).* Οι κλάσεις αναπαριστούν τις έννοιες. Μία έννοια μπορεί να χαρακτηριστεί οτιδήποτε για το οποίο μπορεί να ειπωθεί κάτι, όπως για παράδειγμα μια υπηρεσία ή περιγραφή μίας διαδικασίας. Οι έννοιες μπορούν να διακριθούν σε *πρωταρχικές* και σε *οριζόμενες* έννοιες. Πρωταρχικές έννοιες είναι αφηρημένες κλάσεις που δεν χαρακτηρίζονται από κάποιες συγκεκριμένες ιδιότητες όπως για παράδειγμα η κλάση «Άνθρωπος», σε αντίθεση με τις οριζόμενες έννοιες που δεν είναι αφηρημένες κλάσεις και χαρακτηρίζονται από συγκεκριμένες ιδιότητες. Οι οντολογίες που χρησιμοποιούν οριζόμενες έννοιες περιέχουν κλάσεις οι οποίες είναι τελείως διαφορετικές μεταξύ τους, ενώ στις οντολογίες που χρησιμοποιούνται πρωταρχικές έννοιες ως κλάσεις, μία κλάση είναι υποκλάση ή υπερκλάση μία άλλης.

- *Στιγμιότυπα (Individuals).* Τα στιγμιότυπα είναι αντικείμενα που αναπαριστούν στοιχεία και ανήκουν σε μία κλάση.

- *Σχέσεις (Relations).* Εκφράζουν ουσιαστικά τους τρόπους με τους οποίους μπορούν να αλληλεπιδρούν μεταξύ τους οι κλάσεις και τα στιγμιότυπα μια οντολογίας. Σε μία οντολογία μπορούμε να έχουμε σχέσεις μεταξύ κλάσεων , σχέσεις μεταξύ στιγμιότυπων, σχέσεις μεταξύ κλάσεων και στιγμιότυπων καθώς επίσης και σχέσεις μεταξύ στιγμιότυπων και λεκτικών αντικειμένων. Για παράδειγμα, στην σχέση «Ο Κώστας έχει ύψος 1.80», ο Κώστας είναι στιγμιότυπο της κλάσης «Άνθρωπος», η σχέση είναι η φράση «έχει ύψος» και το «1.80» είναι το λεκτικό αντικείμενο.

- *Αξιώματα (Axioms).* Τα αξιώματα χρησιμοποιούνται για την έκφραση προτάσεων που είναι πάντοτε αληθείς. Για παράδειγμα αν το αυτοκίνητο «Opel Corsa» είναι μικρό αυτοκίνητο τότε μπορεί να ενταχθεί στην κατηγορία μικρά αυτοκίνητα.

- *Συναρτήσεις (Functions).* Πρόκειται ουσιαστικά για έναν συνδυασμό σχέσεων, όπου το ν-οστό στοιχείο της σχέσης προσδιορίζεται μοναδικά από τα ν-1 προηγούμενα στοιχεία.

2.2.3 *Βασικες Αρχες Υλοποιηση Οντολογιας*

Στο σημείο αυτό κρίνεται απαραίτητο να πραγματοποιηθεί μία αναφορά σχετικά με τις αρχές και του κανόνες που θα πρέπει να λαμβάνονται υπόψιν κατά την υλοποίηση μιας οντολογίας. Μία οντολογία θα πρέπει να χαρακτηρίζεται από[16]:

- *Σαφήνεια (Clarity).* Σε μία οντολογία η οποία υλοποιείται για την περιγραφή μιας συγκεκριμένης γνωστικής περιοχής, θα πρέπει τα στοιχεία της οντολογίας να αποδίδονται με όσο το δυνατόν πιο σαφή τρόπο.

- *Συνοχή (Coherence).* Μια οντολογία θα πρέπει να είναι συνεπής δηλαδή μέσα από τις προτάσεις που περιγράφονται σε μία οντολογία θα πρέπει να εξάγονται συμπεράσματα τα οποία να μην έρχονται σε αντίθεση μεταξύ τους.

- *Επεκτασιμότητα (Extendibility).* Μία οντολογία θα πρέπει να έχει σαν έναν από τους κυρίους στόχους της τον διαμοιρασμό της, οπότε θα πρέπει να είναι όσο το δυνατόν επεκτάσιμη. Για αυτόν τον λόγο θα πρέπει να χρησιμοποιείται για την υλοποίηση της κοινό λεξιλόγιο ως προς άλλες οντολογίες για να μπορεί να επεκταθεί στην συνέχεια με μεγαλύτερη ευκολία, μιας και είναι δύσκολο να αποδοθούν από την αρχή επακριβώς όλες οι πληροφορίες μίας γνωστικής περιοχής.

- *Ελάχιστο εύρος κωδικοποίησης (Minimal Encoding bias).* Η κωδικοποίηση της οντολογίας, θα πρέπει να γίνεται με τον απλούστερο δυνατό τρόπο, απαιτώντας όσο το δυνατόν μικρότερους και απλούστερους ορισμούς για την αναπαράστασή της.

- *Ελάχιστη οντολογική δέσμευση (Minimal Ontological Commitment).* Τα αξιώματα που εμπεριέχονται στην οντολογία πρέπει να είναι γενικευμένα. Πρέπει να αποφεύγεται η σαφής αναφορά στο πεδίο ενδιαφέροντος που μοντελοποιείται, έτσι ώστε να υπάρχει ελευθερία για εξειδίκευση και εφαρμογή σε συγκεκριμένες γνωστικές περιοχές καθώς και για επαναχρησιμοποίηση της οντολογίας σε άλλα πεδία ενδιαφέροντος.

2.3 Επισκοπηση

Στις ενότητες που ακολουθούν περιγράφονται αρχικά οι κυριότερες γλώσσες που χρησιμοποιούνται για την αναπαράσταση και την περιγραφή σημασιολογικά των δεδομένων που αποτελούν μία οντολογία. Στην συνέχεια εισερχόμαστε στην ενότητα της Ασάφειας και της Αβεβαιότητας που σχετίζονται με την αναπαράσταση και την περιγραφή σημασιολογικά ασαφών δεδομένων κατά συνέπεια ασαφών οντολογιών. Τέλος

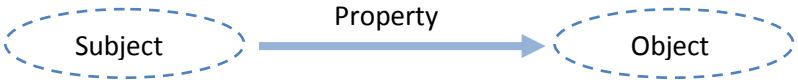
πραγματοποιείται μια συνοπτική περιγραφή της σημασιολογικής γλώσσας περιγραφής ασάφειας «f-OWL».

2.4 Γλωσσες Σημασιολογικης Περιγραφης

2.4.1 *Η Γλωσσα RDF*

Η γλώσσα RDF (Resource Description Framework) είναι μία περιγραφική γλώσσα βασισμένη στην XML η οποία χρησιμοποιείται για την απλή περιγραφή πόρων του διαδικτύου. Η περιγραφή αυτή εστιάζεται στην απόδοση μετα-πληροφορίας για τις οντότητες αυτές, όπως είναι η περιγραφή του τίτλου, του ονόματος, της ημερομηνίας δημιουργίας αλλά και άλλων πόρων του διαδικτύου. Με τον όρο πόρο αναφερόμαστε σε οποιαδήποτε οντότητα του παγκόσμιου ιστού, όπως είναι μία ιστοσελίδα, ένα τμήμα ή ένα σύνολο από ιστοσελίδες, ηλεκτρονικά αρχεία ή ακόμα αντικείμενα τα οποία δεν είναι άμεσα διαθέσιμα στο διαδίκτυο (π.χ. ένα υπαρκτό πρόσωπο).

Με τον όρο μετα – πληροφορία αναφερόμαστε στην ιδιότητα ενός πόρου του διαδικτύου και στην τιμή της ιδιότητα που έχει ο πόρος αυτός. Ουσιαστικά αυτό αποτελεί και μία RDF πρόταση (sentence). Μία RDF πρόταση εκφράζεται σαν μία τριάδα (triple) στοιχείων, αποτελούμενη από ένα υποκείμενο (subject) το οποίο καταλαμβάνεται από έναν πόρο, από μία ιδιότητα (property ή predicate) και από ένα αντικείμενο (object) στο οποίο αντιστοιχεί η τιμή της ιδιότητας αυτής για τον πόρο αυτό. Η τιμή της ιδιότητας αυτής μπορεί να είναι είτε ένας άλλος πόρος είτε απλώς κάποια δεδομένα. Μία RDF πρόταση μπορεί να περιγραφεί και σαν ένας γράφος (εικόνα 2.2), όπου οι κόμβοι αντιστοιχούν είτε σε υποκείμενα είτε σε αντικείμενα ενώ οι ακμές αντιστοιχούν σε ιδιότητες.



Εικόνα 2.2 – Απεικόνιση RDF Γράφου

Η κατεύθυνση της ακμής είναι πολύ σημαντική γιατί μας δείχνει πάντα το αντικείμενο.

Συγκεκριμένα:

- Ένας κόμβος μπορεί να είναι ένα Uri με ή χωρίς προσδιοριστικά ή ένα λεκτικό (Literal).

- Σε περίπτωση που το Uri περιέχει κάποιο προσδιοριστικό για παράδειγμα www.uop.gr#subject1 , το #subject1 αποτελεί ένα Fragment Identifier.

- Μία Ιδιότητα εκφράζεται ως ένα Uri και αντιστοιχεί σε μία ακμή.

Πριν ξεκινήσουμε να αναφερόμαστε στην σύνταξη της RDF, κρίνεται σκόπιμο η παρουσίαση ενός παραδείγματος απόδοσης μετα-πληροφορίας. Έστω ότι επιθυμούμε να περιγράψουμε ότι η ιστοσελίδα www.uop.gr δημιουργήθηκε στις 02/03/2002 , η RDF πρόταση που προκύπτει από τον παραπάνω συλλογισμό εκφράζεται με την ακόλουθη τριάδα.

http://www.uop.gr/index.html http://www.uop.gr/creation-date ”02-02-2002”.

Subject Property Object (Τύπου Literal)

Η τιμή της ιδιότητας www.uop.gr/creation-date είναι το λεκτικό 02/03/2002 το οποίο αποτελεί και το αντικείμενο της τριάδας. Είναι πολύ σημαντικό να τονίσουμε ότι ένα λεκτικό δεν μπορεί να είναι στην θέση του υποκείμενου ή της ιδιότητας. Μας δίνεται όμως η δυνατότητα να ορίσουμε τι τύπου μπορεί να είναι ένα λεκτικό, χρησιμοποιώντας τους τύπους δεδομένων του XML Schema Definition (εικόνα 2.3 [4]). Επειδή το λεκτικό 02/03/2002 αποτελεί ημερομηνία θα μπορούσε να δηλωθεί με τον εξής τρόπο: “02/03/2002”^^xsd:date. Με αυτό τον τρόπο δηλώνουμε ότι το συγκεκριμένο λεκτικό είναι τύπου ημερομηνία (date). Η συντόμευση xsd αντιστοιχεί στο Uri http://www.w3.org/2001/XMLSchema#.

Για την καλύτερη διαχείριση των Uri μπορούμε να χρησιμοποιήσουμε συντομεύσεις σε μορφή προθεμάτων, τα οποία ονομάζονται Qnames. Qname αποτελεί και η συντόμευση xsd που αναφέραμε παραπάνω. Για να γίνει πιο κατανοητή η χρήση των Qnames, θα αναπαραστήσουμε στο παράδειγμα μας το Uri http://www.uop.gr με την συντόμευση – Qname «uop». Από την παραπάνω αλλαγή προκύπτει η ακόλουθη τριάδα:

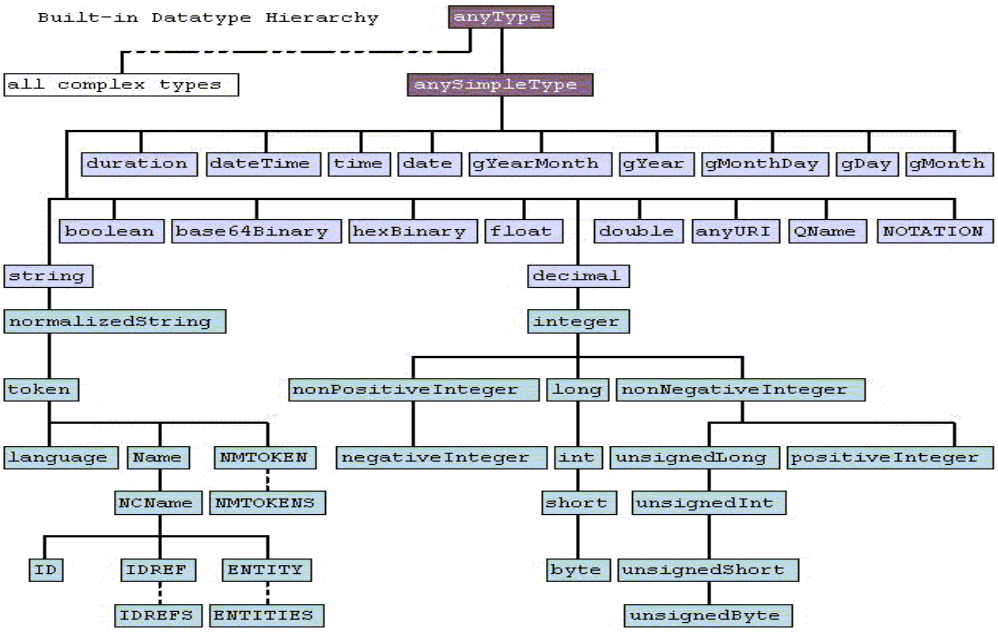
uop:index.html uop:creation-date ”02-02-2002”.

Subject Property Object (Τύπου Literal)

Στον σημασιολογικό ιστό χρησιμοποιούνται συχνά συγκεκριμένα Qnames, μερικά από τα όποια παρουσιάζονται στον πίνακα 2.1. Η απεικόνιση της παραπάνω τριάδας σε RDF γράφο απεικονίζεται στην εικόνα 2.4.

Πρόθεμα (prefix)	Namespace URI (location)
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns
rdfs	http://www.w3.org/2000/01/rdf-schema
xsd	http://www.w3.org/2001/XMLSchema
dc	http://purl.org/dc/elements/L1/
daml	http://www.daml.org/2001/03/daml+oil
owl	http://www.w3.org/2002/07/owl
cc	http://creativecommons.org/ns
rdfa	http://www.w3.org/ns/rdfa
foaf	http://xmlns.com/foaf/0.1
swrl	http://www.w3.org/2003/11/swrl

Πίνακας 2.1 – Συνήθη Qnames στο Σημασιολογικό Ιστό



Εικόνα 2.3 – Η ιεραρχία XML Datatypes

Εικόνα 2.4 – RDF γράφος

Στην συνέχεια παραθέτουμε την διατύπωση της παραπάνω τριάδας σε RDF γλώσσα.

```
<?xml version="1.0"?>
 <rdf:RDF  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:uop="http://www.uop.gr/"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
       <rdf:Description rdf:about="http://www.uop.gr/">
           <uop:creation-date rdf:datatype="xsd;date">02-02-2002</uop:creation-date>
       </rdf:Description>
</rdf:RDF>
```

Πίνακας 2.2 – RDF κώδικας

Όπως μπορούμε να διακρίνουμε από τον πίνακα 2.2 ο RDF κώδικας ξεκινάει πάντα με την δήλωση *<?xml version="1.0"?>* με την οποία δηλώνουμε ότι πρόκειται για xml έγγραφο. Στην συνέχεια με την δήλωση *rdf:RDF* δηλώνουμε ότι θα ακολουθήσει ένα σύνολο από RDF προτάσεις. Πριν αρχίσουμε να διατυπώνουμε μία τριάδα ορίζουμε τα Qnames τα οποία επιθυμούμε (στο παράδειγμα μας «uop», «xsd» και «rdf»). Μόλις ορίσουμε τα Qnames είμαστε σε θέση να αποδώσουμε μετά – πληροφορία στους πόρους που

επιθυμούμε. Ξεκινώντας με το στοιχείο *rdf:Description* υποδηλώνουμε την αρχή της περιγραφής ενός χαρακτηριστικού ενός πόρου. Στην συνέχεια με το στοιχείο *rdf:about* δηλώνουμε σε ποιον πόρο αναφερόμαστε όπου ο συγκεκριμένος πόρος θα αποτελέσει και το υποκείμενο της πρότασης μας. Κατόπιν μπορούμε να προβούμε στην δήλωση των ιδιοτήτων. Συγκεκριμένα στο παράδειγμα μας έχουμε δηλώσει ότι για την ιδιότητα *creation-date* η τιμή της είναι 02-02-2002 και βρίσκεται μεταξύ των ετικετών εκκίνησης και τέλους της ιδιότητας *«creation-date».* Αυτή η τιμή θα αποτελέσει και το αντικείμενο της πρότασης μας. Με το στοιχείο *rdf:datatype="xsd;date"* δηλώνουμε ότι αυτό που θα ακολουθήσει είναι λεκτικό και συγκεκριμένα δηλώνουμε ότι είναι τύπου ημερομηνίας.

Σε περίπτωση που επιθυμούσαμε να ορίσουμε ως αντικείμενο μιας πρότασης έναν πόρο στην RDF θα το δηλώναμε έως εξής:

```
<?xml version="1.0"?>
 <rdf:RDF  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:uop="http://www.uop.gr/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
      <rdf:Description rdf:about="http://www.uop.gr/">
          <uop:creation-date rdf:datatype="xsd;date">02-02-2002</uop:creation-date>
          <uop:creator rdf:resource="http://www.uop.gr#creator1"/>
      </rdf:Description>
</rdf:RDF>
```

Πίνακας 2.3 RDF κώδικας ορισμού αντικειμένου ως πόρου

Με την δήλωση *<uop:creator rdf:resource="http://www.uop.gr#creator1"/>* ορίζουμε ότι για την ιδιότητα uop:creator η τιμή της είναι ένα αντικείμενο που είναι πόρος και αυτό το επιτυγχάνουμε με την δήλωση *rdf:resource=http://www.uop.gr#creator1.* Είναι πολύ σημαντικό αναφέρουμε στο σημείο αυτό, ότι δεν μπορούμε να ορίσουμε κάποιο Qname ως τίμη κάποιας ιδιότητας, θα πρέπει να ορίσουμε ολόκληρο το Uri. Αυτό ακριβώς αποτελεί και περιορισμό από το διεθνές πρότυπο της RDF[5].

Από τα παραπάνω μπορούμε να συμπεράνουμε ότι από ένα μεγάλο σύνολο RDF αποφάσεων, μπορεί να προκύψει μια αλυσίδα, όπου το υποκείμενο της μίας πρότασης να είναι αντικείμενο κάποιας άλλης πρότασης.

Στο σημείο αυτό θα αναφερθούμε στον όρο του «Κενού Κόμβου» (Blank Node)[5]. Στο παράδειγμα που περιγράψαμε θα μπορούσαμε να αναλύσουμε την ημερομηνία ως έναν επιπλέον πόρο με την ονομασία «uop:dateid_1» περιγράφοντας τρείς ιδιότητες για τον πόρο αυτό , ημέρα, μήνας και χρόνος. Ακολούθως θα είχαμε τις παρακάτω τριάδες:

uop:index.html uop:creation-date uop: dateid_1.
uop:dateid_1 uop:day 02.
uop:dateid_1 uop:month 02.
uop:dateid_1 uop:year 2002.

Στην παραπάνω περίπτωση μπορούμε να θεωρήσουμε ότι ο πόρος «uop:dateid_1» δεν έχει κάποια ιδιαίτερη χρησιμότητα οπότε θα μπορούσαμε να εισάγουμε έναν πόρο χωρίς να του ορίσουμε κάποιο συγκεκριμένο Uri. Ένας πόρος στον οποίο δεν αντιστοιχεί κάποιο

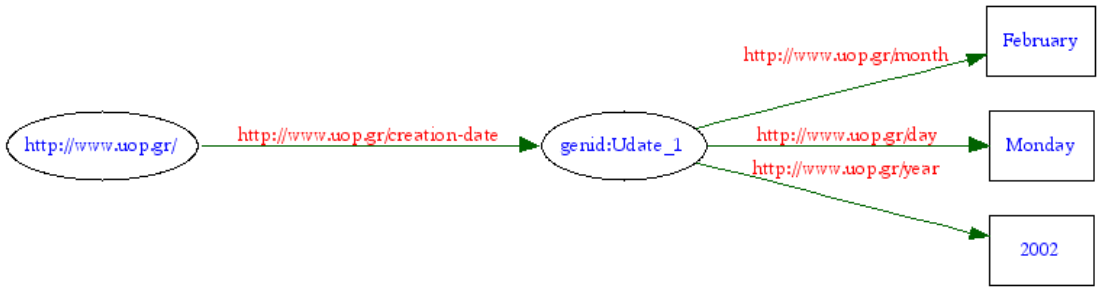
Uri ονομάζεται Κενός Κόμβος. Σε ένα RDF έγγραφο που περιγράφει μία οντολογία θα μπορεί να συνυπάρχουν πολλοί Κενοί Κομβόι. Για την καλύτερη διαχείριση τους αλλά και κατανόησης τους μας δίνεται η ευχέρεια να χρησιμοποιήσουμε τους «Προσδιοριστές Κενών Κόμβων» (Blank Nodes Identifiers), με το να αντιστοιχήσουμε ένα ξεχωριστό όνομα για τον κάθε κενό κόμβο. Συνεπώς, οι αντίστοιχες τριάδες που προκύπτουν στο παράδειγμα μας, από την χρησιμοποίηση Κενού κόμβου στην θέση του πόρου «uop:dateid_1» είναι οι εξής:

uop:index.html uop:creation-date _: date_1.
_: date_1 uop:day 02.
_: date_1 uop:month 02.
_: date_1 uop:year 2002.

Με το στοιχείο «_:» δηλώνουμε την ύπαρξη Κενού Κόμβου ενώ στην συνέχεια δηλώνουμε ένα αναγνωριστικό (date_1) για τον συγκεκριμένο κόμβο. Στον πίνακα 2.4 και στην εικόνα 2.4 παραθέτουμε τον κώδικα RDF για τις παραπάνω τριάδες και τον RDF γράφο αντίστοιχα.

```
<?xml version="1.0"?>
 <rdf:RDF  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:uop="http://www.uop.gr/"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
                    <rdf:Description rdf:about="http://www.uop.gr/" >
         <uop:creation-date rdf:nodeID="date_1"/>
                    </rdf:Description>
                    <rdf:Description rdf:nodeID="date_1">
                              <uop:month>February</uop:month>
                              <uop:day>Monday</uop:day>
                              <uop:year>2002</uop:year>
             </rdf:Description>
            </rdf:RDF>
```

Πίνακας 2.4 - RDF κώδικας με την χρησιμοποίηση Κενού Κόμβου



Εικόνα 2.5 – Απεικόνιση Κενού Κόμβου σε RDF Γράφο

Για να δηλώσουμε στην RDF ότι ένας πόρος είναι στιγμιότυπο κάποιας κλάσης θα πρέπει να χρησιμοποιήσουμε το στοιχείο rdf:type. Ένας πόρος μπορεί να επενεργεί ταυτόχρονα και ως κλάση (δηλαδή έννοια) αλλά και μεμονωμένα ως άτομο.

Σε ορισμένες περιπτώσεις κατά την δημιουργία μιας οντολογίας, χρειαζόμαστε να δημιουργήσουμε προτάσεις οι οποίες αναφέρονται σε άλλες προτάσεις. Έτσι για παράδειγμα μπορούμε να περιγράψουμε το πότε δημιουργήθηκε κάποια τριάδα και κατά

συνέπεια μία πρόταση ή το ποιος δημιούργησε την συγκεκριμένη πρόταση. Το λεξιλόγιο της RDF παρέχει τα απαραίτητα στοιχεία για την δημιουργία τέτοιου είδους προτάσεων. Η διαδικασία δημιουργίας προτάσεων με την χρήση του λεξιλογίου αυτού ονομάζεται «Reification». Παρακάτω περιγράφουμε τα στοιχεία που ανήκουν στο λεξιλόγιο αυτό καθώς και ένα παράδειγμα χρήσης τους (πίνακας 2.5)[5]:

- rdf:Statement – Δηλώνουμε ότι ένας πόρος αποτελεί μία πρόταση – μία τριάδα.
- rdf:subject – Δηλώνουμε τον πόρο που αντιστοιχεί σε ρόλο υποκείμενου σε ένα statement.
- rdf:object – Δηλώνουμε τον πόρο ή το λεκτικό που αντιστοιχεί σε ρόλο αντικείμενου σε ένα statement.
- rdf:predicate – Δηλώνουμε τον πόρο που αντιστοιχεί σε μία ιδιότητα σε ένα statement.

```
uop:triple12345   rdf:type       rdf:Statement .
uop:triple12345   rdf:subject    uop:index.html .
uop:triple12345   rdf:predicate  uop:creation-date .
uop:triple12345   rdf:object     "02-02-2002"^^xsd:date .
```

Πίνακας 2.5 – Παράδειγμα χρήσης Reification

Ολοκληρώνοντας, η RDF γλώσσα μας παρέχει επιπλέον στοιχεία με τα οποία μπορούμε να ορίσουμε και να περιγράψουμε σύνολα (bags), ακολουθίες (sequences) και σύνολο από εναλλακτικές επιλογές (alternatives). Το RDF λεξιλόγιο που παρέχετε για τον ορισμό αυτών των δομών είναι τα εξής στοιχεία:

- rdf:bag (Σύνολο)
- rdf:seq (Ακολουθία)
- rdf: alt (Σύνολο Εναλλακτικών Επιλογών)

2.4.2 *Το Σχημα RDF (RFD SCHEMA)*

Η RDF γλώσσα μας παρέχει ένα λεξιλόγιο με το οποίο μπορούμε αποδώσουμε μετα-πληροφορία σε έναν πόρο του διαδικτύου. Δεν μας δίνει όμως την δυνατότητα να ορίσουμε κλάσεις (έννοιες) στα πλαίσια περιγραφής μιας οντολογίας ενός συγκεκριμένου πεδίου εφαρμογής ούτε να ορίσουμε σχέσεις υπαγωγής μεταξύ κλάσεων αλλά και ιδιοτήτων. Αυτή την έλλειψη έρχεται να καλύψει ένα επιπλέον λεξιλόγιο για την RDF, το «RDF SCHEMA». Κύριος στόχος του RDF SCHEMA είναι να βάλει σε μια τάξη την σημασιολογία του RDF αλλά και να δημιουργήσει μια ιεραρχία μεταξύ των RDF προτάσεων.

Στον πίνακα 2.6 παρουσιάζουμε την RDFS σύνταξή. Το βασικό λεξιλόγιο καθορίζεται με ένα Qname που εδώ καλείται «rdfs». Αυτό προσδιορίζεται από το URI http://www.w3.org/2000/01/rdf-schema# .

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xml:base="http://www.test.gr/schema/">

<rdfs:Class rdf:ID="Person"/>

<rdfs:Class rdf:ID="Player">
<rdfs:subClassOf rdf:resource="#Person"/>
</rdfs:Class>

<rdfs:Class rdf:ID="HeadDoctor">
<rdfs:subClassOf rdf:resource="#Person"/>
</rdfs:Class>


<rdf:Property rdf:ID="health_responsible">
<rdfs:domain rdf:resource="#HeadDoctor"/>
<rdfs:range rdf:resource="#Player"/>
<rdfs:comment>HeadDoctor is responsible for the health of players</rdfs:comment>
</rdf:Property>

</rdf:RDF>
```

Πίνακας 2.6 – RDF SCHEMA

Παρακάτω περιγράφουμε τα στοιχεία του λεξιλογίου RDF SCHEMA, πολλά από τα οποία έχουνε χρησιμοποιηθεί και στο παράδειγμα του πίνακα 2.6:

- rdfs:Class – Χρησιμοποιείται για να δηλώσουμε μία κλάση.
- rdfs:subClassOf – Χρησιμοποιείται για να δηλώσουμε μία υποκλάση. Με το συγκεκριμένο στοιχείο εκφράζουμε μία σχέση υπαγωγής μεταξύ κλάσεων.
- rdfs:domain – Χρησιμοποιείται για να δηλώσουμε ότι πεδίο ορισμού μιας ιδιότητας (το υποκείμενο σε μία πρόταση) θα πρέπει να είναι στιγμιότυπο μιας συγκεκριμένης κλάσης.
- rdfs:range – Χρησιμοποιείται για να δηλώσουμε ότι πεδίο τιμών μιας ιδιότητας (το αντικείμενο σε μία πρόταση) θα πρέπει να είναι στιγμιότυπο μιας συγκεκριμένης κλάσης.
- rdfs: comment – Χρησιμοποιείται για να περιγράψουμε μία ιδιότητα ή ένα υποκείμενο ή ένα αντικείμενο μιας πρότασης.
- rdfs: label – Χρησιμοποιείται για να ορίσουμε μία ετικέτα σε μία ιδιότητα ή σε ένα υποκείμενο ή σε ένα αντικείμενο μιας πρότασης.
- rdfs: subPropertyOf – Χρησιμοποιείται για να δηλώσουμε μία υπο-ιδιότητα. Με το συγκεκριμένο στοιχείο εκφράζουμε μία σχέση υπαγωγής μεταξύ ιδιοτήτων.

Από τον συνδυασμό της RDF και του RDF SCHEMA και των σχέσεων υπαγωγής που μπορούμε να δηλώσουμε μεταξύ κλάσεων αλλά και ιδιοτήτων, μπορούμε να προβούμε στην εξαγωγή πολύ σημαντικών σημασιολογικών συμπερασμάτων. Στον πίνακα 2.6 έχουμε δηλώσει μία σχέση υπαγωγής μεταξύ των κλάσεων «Person» και «Player», όπου ή κλάση Player αποτελεί υποκλάση της κλάσης Person. Στον πίνακα 2.7 που ακλουθεί, περιγράφουμε έναν πόρο χρησιμοποιώντας το RDF SCHEMA του πίνακα 2.6 και δηλώνοντας ότι ο συγκεκριμένος πόρος είναι στιγμιότυπο της κλάσης Player. Το σημασιολογικό συμπέρασμα που προκύπτει από τα παραπάνω, είναι ότι ο πόρος «http://http:www.olympiacos.gr/players#id_001» είναι και στιγμιότυπο τύπου Person αφού η κλάση Player είναι υποκλάση της Person.

```
<rdf:Description rdf:about="http://http:www.olympiacos.gr/players#id_001">
                <exterms:name rdf:datatype="&xsd;string">Antonis Nikopolidis</exterms:name>
                <exterms:datebirth rdf:datatype="&xsd;date">1971-08-16</exterms:datebirth>
                <exterms:weight rdf:datatype="&xsd;decimal">70.5</exterms:weight>
                <exterms:height rdf:datatype="&xsd;decimal">1.80</exterms:height>
                <exterms:debut rdf:datatype="&xsd;date">1990-08-16</exterms:debut>
               <exterms:nationality rdf:datatype="&xsd;string">Hellenic</exterms:nationality>
              <exterms:international_participations   rdf:datatype="&xsd;integer">
                90</exterms:international_participations>
              <exterms:loc_participations rdf:datatype="&xsd;integer">200</exterms:loc_participations>
             <exterms:previous_team rdf:resource="http://www.pao.gr/" />
            <exterms:member_plys rdf:resource="http://http:www.olympiacos.gr/players/" />
             <rdf:type>
                    <rdf:Description rdf:about="http://www.w3.org/2000/01/rdf-schema#Player"/>
           </rdf:type>
           </rdf:Description>
```

Πίνακας 2.7 – Περιγραφή ενός πόρου με την γλώσσα RDF

2.4.3 *Η ΓΛΩΣΣΑ OWL*

Οι γλώσσες «RDF» και «RDF SCHEMA» που περιγράψαμε στις προηγούμενες ενότητες χαρακτηρίζονται για την περιορισμένη εκφραστική τους ικανότητα[7]. Με την χρήση της RDF και του RDF SCHEMA μπορούμε να προβούμε στον ορισμό:

- Στιγμιότυπων κλάσεων και ιδιοτήτων.
- Ιεραρχίες κλάσεων και ιδιοτήτων.
- Περιορισμούς που σχετίζονται με το πεδίο ορισμού (rdfs:domain) και πεδίο τιμών (rdfs:range) ιδιοτήτων.

Οι ελλείψεις και οι περιορισμοί που εντοπίζονται από την χρήση της RDF και του RDF SCHEMA συνοψίζονται από τα παρακάτω χαρακτηριστικά:

- Μη δυνατότητα ορισμού τοπικής εμβέλειας ιδιοτήτων.
- Έλλειψη δυνατότητας δημιουργίας νέων κλάσεων από των συνδυασμό υπαρχουσών, δηλαδή νέες κλάσεις που να προκύπτουν από ενώσεις(union), τομές(intersection) ή συμπληρώματα(complement) άλλων κλάσεων.

- Δεν είναι δυνατόν να οριστεί περιορισμός στο πλήθος τιμών του πεδίου τιμών μιας ιδιότητας, για παράδειγμα δεν είναι δυνατό να οριστεί η σχέση, ότι ένα παιδί έχει ακριβώς μια βιολογική μητέρα.

- Δεν υπάρχει δυνατότητα δήλωσης ειδικών χαρακτηριστικών για τις ιδιότητες, για παράδειγμα μία ιδιότητα να είναι μοναδική ή αντίστροφή μιας άλλης.

Ο συνδυασμός των παραπάνω ελλείψεων οδήγησαν διάφορες ερευνητικές ομάδες και την Web Ontology Working Group της W3C στην καθιέρωση μιας νέας σημασιολογικής γλώσσας την «OWL» (Ontology Web Language). Η OWL, καλύπτει όλες τις παραπάνω αδυναμίες, διαθέτει μεγαλύτερη εκφραστικότητα η οποία σαφώς είναι αναγκαία για το Σημασιολογικό Ιστό, καθώς επιτρέπει την καλύτερη διαχείριση και εξαγωγή σημασιολογικών συμπερασμάτων. Αναφερόμενοι στην διατυπωμένη από τον Tim Berners-Lee αρχιτεκτονική του Σημασιολογικού Ιστού, η OWL αντιστοιχεί στο λογικό επίπεδο, το οποίο επιτρέπει την επεξεργασία δεδομένων κατώτερου επιπέδου τα οποία πλέον είναι φορτισμένα σημασιολογικά.

Οι βασικές αρχές της OWL βασίζονται στην «Περιγραφική Λογική»[7] η οποία με την σειρά της βασίζεται στον Κατηγορηματικό Λογισμό. Η OWL εμφανίζεται σε τρείς διαφορετικές εκδοχές:

- Η *OWL Lite* έχει σχεδιαστεί για την έκφραση ιεραρχιών ταξινόμησης και απλών περιορισμών ιδιοτήτων. Για παράδειγμα, ενώ η OWL Lite υποστηρίζει περιορισμούς πληθυκότητας, οι μόνες τιμές που επιτρέπονται είναι 0 και 1. Είναι πιο εύκολο να σχεδιαστούν εργαλεία και να αντιστοιχιστούν θησαυροί όρων και ταξινομίες στην OWL Lite από ότι στα άλλα εκφραστικότερα επίπεδα και είναι πιο εύκολη στην εκμάθηση από τους χρήστες. Το σημαντικότερο μειονέκτημα της είναι η περιορισμένη εκφραστική ικανότητα.

- Η *OWL DL* σχεδιάστηκε για τους χρήστες που επιθυμούν τη μέγιστη δυνατή εκφραστικότητα διατηρώντας:

  - Την υπολογιστική πληρότητα, δηλαδή όλα τα συμπεράσματα να είναι εγγυημένα υπολογίσιμα.

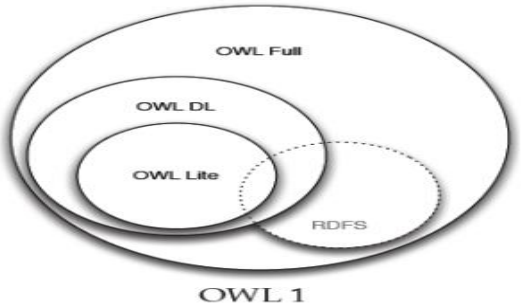
  - Την αποφασισιμότητα, δηλαδή η διεξαγωγή σημασιολογικών συμπερασμών να πραγματοποιείται σε πεπερασμένο χρόνο.

  Η OWL DL ονομάζεται έτσι λόγω της αντιστοιχίας της με την γλώσσα της Περιγραφικής Λογικής και συγκεκριμένα βασίζεται στην γλώσσα περιγραφικής λογικής SHOIN(D). Η OWL DL διευρύνει σημαντικά την OWL Lite από άποψη εκφραστικής ικανότητας. Το υπολογιστικό κόστος που σχετίζεται με την εκφραστική επέκταση είναι σημαντικό, ωστόσο πρόκειται για την πιο ευρέως διαδεδομένη γλώσσα σημασιολογικής περιγραφής.

- Η OWL FULL αποτελεί την πλέον εμπλουτισμένη εκφραστικά εκδοχή της γλώσσας της οποίας η OWL DL είναι υποσύνολο. Περιλαμβάνει στοιχεία της RDF και του RDF SCHEMA. Απευθύνεται αποκλειστικά σε χρήστες που επιθυμούν μέγιστη εκφραστικότητα. Βεβαία, οι ανεξέλεγκτες όμως αυτές εκφραστικές δυνατότητες την καθιστούν μη–αποφασίσιμη και κατά επέκταση μη χρησιμοποιήσιμη σε πολλές εφαρμογές.

Κάθε μία από τις εκδοχές της OWL που αναφέραμε παραπάνω, είναι επέκταση του απλούστερου προκατόχου της και σχηματικά αυτό φαίνεται στην εικόνα 2.6 , από την οποία κιόλας προκύπτουν και οι ακόλουθες ακριβείς σχέσεις:

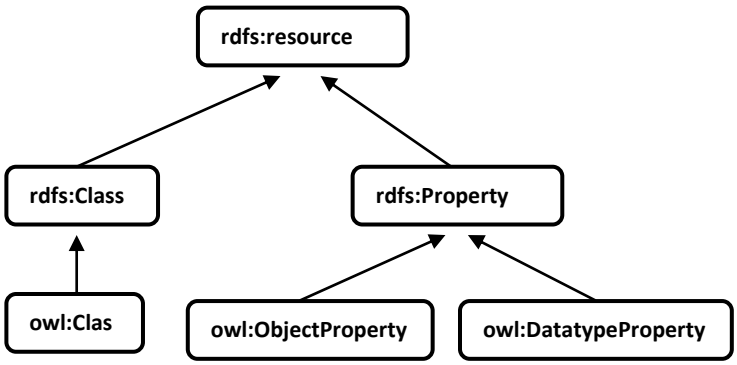
- Κάθε έγκυρη οντολογία της OWL Lite είναι μια έγκυρη οντολογία της OWL DL.
- Κάθε έγκυρη οντολογία της OWL DL είναι μια έγκυρη οντολογία της OWL Full.
- Κάθε έγκυρο συμπέρασμα της OWL Lite είναι ένα έγκυρο συμπέρασμα της OWL DL.
- Κάθε έγκυρο συμπέρασμα OWL DL είναι ένα έγκυρο συμπέρασμα της OWL Full.



Εικόνα 2.6 – Οι εκδοχές της OWL

Όπως αναφέραμε, η OWL αποτελεί μια σημασιολογική γλώσσα που αποσκοπεί στην περιγραφή πόρων του διαδικτύου, για αυτό τον λόγο θα πρέπει να διαθέτει μια σύνταξη που να είναι συμβατή με την XML. Αυτό επιτυγχάνεται με το να εξακολουθεί να χρησιμοποίει την RDF και το RDF SCHEMA σε μεγάλο βαθμό. Συγκεκριμένα:

- Όλα τα είδη της OWL χρησιμοποιούν την RDF για τη σύνταξή τους.
- Τα στιγμιότυπα ορίζονται όπως στην RDF, χρησιμοποιώντας τις RDF περιγραφές και εισάγοντας την πληροφορία.
- Constructors της OWL όπως owl:Class owl:DatatypeProperty και owl:ObjectProperty αποτελούν εξειδικεύσεις των αντίστοιχων της RDF (εικόνα 2.7).



Εικόνα 2.7– Σχέση κλάσης υποκλάσης μεταξύ RDFS και OWL

Η επιλογή μιας εκδοχής της OWL για την ανάπτυξη μιας οντολογίας εξαρτάται από τις ανάγκες υλοποίησης της κάθε οντολογίας και τον βαθμό της εκφραστικής ικανότητας αλλά και την αποφασισιμότητα που απαιτείται να έχει η οντολογία.

Η OWL Full μπορεί να θεωρηθεί ως επέκταση της RDF, ενώ η OWL Lite και η OWL DL ως επεκτάσεις μιας περιορισμένης άποψης της RDF. Κάθε έγγραφο OWL (Lite, DL, Full) είναι ένα έγγραφο RDF, και κάθε έγγραφο RDF είναι ένα έγγραφο OWL Full, αλλά μόνο μερικά έγγραφα RDF θα είναι έγκυρα έγγραφα OWL Lite ή OWL DL. Για το λόγο αυτό, χρειάζεται προσοχή όταν θέλει ένας χρήστης να θεωρήσει ένα έγγραφο RDF ως έγγραφο OWL. Όταν η εκφραστικότητα της OWL DL ή της OWL Lite κρίνεται κατάλληλη, μερικές προφυλάξεις πρέπει να ληφθούν για να εξασφαλίσουν ότι το αρχικό έγγραφο RDF συμμορφώνεται με τους πρόσθετους περιορισμούς από την OWL DL και την OWL Lite. Μεταξύ των άλλων, κάθε Uri που χρησιμοποιείται σαν όνομα κατηγορίας πρέπει να βεβαιωθεί ρητά ότι είναι του τύπου owl:Class και κάθε αντικείμενο πρέπει να βεβαιωθεί ότι ανήκει τουλάχιστον σε μια κατηγορία.

Ένα OWL έγγραφο ξεκινά με την ετικέτα <rdf>. Στην συνέχεια, προκειμένου να δηλώσουμε ότι αυτό που ακολουθεί είναι μία οντολογία εκφρασμένη σε OWL, χρησιμοποιούμε την ετικέτα owl:Ontology. Η ετικέτα αυτή, περιέχει και το στοιχείο rdf:about με το οποίο μας παρέχεται η δυνατότητα ονομασίας της οντολογίας. Παρακάτω παρουσιάζεται στον πίνακα 2.8, το λεξιλόγιο των στοιχείων της OWL καθώς και παραδείγματα χρήσης τους.

Δηλώσεις επικεφαλίδας

owl:Ontology - Συνήθως επικεφαλίδα των OWL εγγράφων και περιέχει δηλώσεις για τη διαχείριση της οντολογίας, όπως σχόλια, αριθμός έκδοσης και αναφορές σε άλλες οντολογίες.
owl:PriorVersion - Μια αναφορά στην προηγούμενη έκδοση της οντολογίας.
owl:versionInfo - Περιέχει ένα αλφαριθμητικό με πληροφορίες για την παρούσα έκδοση.
owl:backwardCompatibleWith - Περιέχει μια αναφορά στην προηγούμενη έκδοση της οντολογίας η οποία είναι συμβατή με την παρούσα.
owl:incompatibleWith - Το αντίθετο της backwardCompatibleWith. Περιέχει μια αναφορά σε μια οντολογία η οποία δεν είναι συμβατή με την παρούσα. Χρησιμεύει στους

προγραμματιστές ώστε να γνωρίζουν ότι δε μπορούν για παράδειγμα να αναβαθμίσουν μια αναπτυσσόμενη οντολογία χωρίς να ελέγξουν τι μετατροπές χρειάζονται.

owl:Imports - Δίνει λίστα με άλλες οντολογίες των οποίων τα περιεχόμενα θεωρούνται ότι είναι μέρος της παρούσας οντολογίας. Η Imports είναι μεταβατική ιδιότητα, αν δηλαδή η οντολογία *A* εσωκλείει τη *B* και η *B* την *C*, τότε και η *A* εσωκλείει την *C*. Η δήλωση αυτή συμπεριλήφθηκε στην OWL με στόχο η τοποθεσία της κάθε οντολογίας να είναι και το διακριτικό της χαρακτηριστικό.

Παράδειγμα Υλοποίησης

```
<?xml version='1.0' encoding='lSO-8859-1'?>
<rdf:RDF
  xmlns:owl  ="http://www.w3.org/2002/07/owl#"
  xmlns:rdf  ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs ="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xsd  ="http://www.w3.org/2001/XMLSchema#"
 >
<owl:Ontology rdf:about="">
 <owl:versionInfo>$Id$</owl:versionInfo>
 <rdfs:comment>An example OWL ontology</rdfs:comment>
  <owl:priorVersion rdf:resource="http://www.example.org/test-101.owl"/>
  <owl:imports rdf:resource="http://www.example.org/test1.owl"/>
</owl:Ontology>
```

Κλάσεις

owl:Class - Οι κλάσεις της OWL δηλώνονται με χρήση του στοιχείου owl:Class.

owl:equivalentClass - Δήλωση ισοδυναμίας μεταξύ κλάσεων.

owl:disjointWith - Δήλωση ότι μια κλάση δεν έχει κοινά στοιχεία με κάποια άλλη.

Παράδειγμα Υλοποίησης

```
<owl:Class rdf:ID="Animal">
  <rdfs:label>Animal</rdfs:label>
  <rdfs:comment>
    Not vegetable or mineral.
  </rdfs:comment>
  <owl:Class rdf:ID="Male">
    <rdfs:label>Male</rdfs:label>
    <rdfs:subClassOf rdf:resource="#Animal"/>
  </owl:Class>
  <owl:Class rdf:ID="Female">
    <rdfs:label>Female</rdfs:label>
    <rdfs:subClassOf rdf:resource="Animal"/>
    <owl:disjointWith rdf:resource="Male"/>
  </owl:Class>
</owl:Class>
```

Δηλώνουμε ότι οι κλάσεις Male και Female είναι ξένες μεταξύ τους.

Ιδιότητες

owl:ObjectProperty - Ορίζει μια ιδιότητα αντικείμενου η οποία αντιστοιχεί ένα αντικείμενο σε

ένα άλλο.

owl:DatatypeProperty - Ορίζει μια ιδιότητα τύπου δεδομένων η οποία αντιστοιχίζει ένα αντικείμενο σε μία τιμή δεδομένων.

owl:inverseOf - Ορισμός αντίστροφων ιδιοτήτων, όπως για παράδειγμα τις ιδιότητες «προπονείται από» και «προπονεί».

Παράδειγμα Υλοποίησης

```
<owl:ObjectProperty rdf:ID="hasParent">
  <rdfs:domain rdf:resource="#Animal"/>
  <rdfs:range rdf:resource="#Animal"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="hasFather">
  <rdfs:subPropertyOf rdf:resource="#hasParent"/>
  <rdfs:range rdf:resource="#Male"/>
</owl:ObjectProperty>
```

Η ιδιότητα hasFather είναι υπό-ιδιότητα της hasParent και το πεδίο τιμών της θα αντιστοιχεί σε στιγμιότυπα της κλάσης Male.

<u>Περιορισμοί Ιδιοτήτων (Restriction)</u>

owl:Restriction - Περιέχει τους περιορισμούς των ιδιοτήτων. Συγκεκριμένα, περιέχει ένα αντικείμενο onProperty και μία ή περισσότερες δηλώσεις περιορισμών.

owl:onProperty - Ορίζει την ιδιότητα στην οποία εφαρμόζονται οι περιορισμοί .

owl:someValuesFrom - Ορίζει ότι υπάρχει ένα στιγμιότυπο της κλάσης που δηλώνεται το οποίο ικανοποιεί την ιδιότητα.

owl:allValuesFrom - Δηλώνει την κλάση των δυνατών τιμών που μπορεί να πάρει η ιδιότητα που δηλώνεται στην onProperty. Με άλλα λόγια, όλες οι τιμές της ιδιότητας που δηλώνεται στην onProperty πρέπει να είναι από την κλάση που δηλώνεται στην allValuesFrom.

owl:hasValue - Δηλώνει μια συγκεκριμένη τιμή που πρέπει να έχει η ιδιότητα που δηλώνεται με την onProperty.

owl:minCardinality – Δηλώνει τον ελάχιστο αριθμό πληθυκότητας αντικειμένων σε μία ιδιότητα.

owl:maxCardinality – Δηλώνει τον μέγιστό αριθμό πληθυκότητας αντικειμένων σε μία ιδιότητα.

owl:cardinality – Δηλώνει τον ακριβή αριθμό πληθυκότητας αντικειμένων σε μία ιδιότητα.

Παράδειγμα Υλοποίησης

```
<owl:Class rdf:ID="Person">
  <rdfs:subClassOf rdf:resource="#Animal"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasParent"/>
      <owl:toClass rdf:resource="#Person"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction owl:cardinality="1">
      <owl:onProperty rdf:resource="#hasFather"/>
    </owl:Restriction>
  </rdfs:subClassOf>
```

Ένα στιγμιότυπο της κλάσης Person θα μπορεί να έχει ακριβώς έναν πατέρα.

```
</owl:Class>
```

Ειδικές Ιδιότητες

owl:TransitiveProperty - Δηλώνει μια μεταβατική ιδιότητα. Για παραδειγμα η ιδιότητα «είναι προγονός(is_ancestor_of)».

owl:SymmetricProperty - Δηλώνει μια συμμετρική ιδιότητα. Μία ιδιότητα R ονομάζεται συμμετρική αν για κάθε a, b, R(a, b) συνεπάγεται ότι R(b, a). Για παράδειγμα, η ιδιότητα «έχει ίδιο βαθμό με (has same grade as)» είναι συμμετρική.

owl:FunctionalProperty - Δηλώνει μια ιδιότητα είναι συναρτησιακή, δηλαδή ότι έχει το πολύ μια τιμή για κάθε αντικείμενο, όπως η ιδιότητα βάρος «Βάρος (weight)».

owl:InverseFunctionalProperty - Ορίζει μια ιδιότητα για την οποία δύο διαφορετικά αντικείμενα δεν μπορούν να έχουν την ίδια τιμή.

Παράδειγμα Υλοποίησης

```
<owl: DatatypeProperty rdf:ID="hasAge">
  <rdfs:domain rdf:resource="# Person "/>
</owl:ObjectProperty>
<owl:FunctionalProperty rdf:about="# hasAge " />
```

Ορίζουμε ότι η ιδιότητα has Age είναι συναρτησιακή.

Απαρίθμηση

owl:one of - Χρησιμοποιείται για να δηλώσει μια κλάση απαριθμώντας τα στοιχεία που την αποτελούν.

Παράδειγμα Υλοποίησης [7]

```
<owl:Class rdf:ID="weekdays">
<owl:oneOf rdf:parseType="Collection">
  <owl:Thing rdf:about="#Eurasia"/>
  <owl:Thing rdf:about="#Africa"/>
  <owl:Thing rdf:about="#North_America"/>
  <owl:Thing rdf:about="#South_America"/>
  <owl:Thing rdf:about="#Australia"/>
  <owl:Thing rdf:about="#Antarctica"/>
</oneOf>
</owl:Class>
```

Ορίζουμε ότι μία κλάση με την ονομασία weekdays καθώς και τα στοιχεία που την αποτελούν.

Λογικοί Συνδυασμοί

owl:unionOf - Ένωση Κλάσεων.

owl:intersectionOf - Τομή Κλάσεων. Σημειώνεται ότι τόσο στην ένωση όσο και στην τομή, η κλάση που προκύπτει δεν είναι υποκλάση του συνόλου που προκύπτει αλλά ισοδύναμη του αποτελέσματος.

owl:ComplementOf - Δηλώνει μια κλάση τα οποία τα στιγμιότυπα της δεν θα είναι ιδία με τα στιγμιότυπα μίας άλλης κλάσης. Παρέχει την ίδια χρησιμότητα με το στοιχείο owl:DisJointClasses.

Παράδειγμα Υλοποίησης

```
<owl:Class rdf:about="#Place">
 <rdfs:subClassOf>
  <owl:Class>
   <owl:complementOfrdf:resource="#Person"/>
  </owl:Class>
 </rdfs:subClassOf>
</owl:Class>
```

Κάθε στιγμιότυπο της κλάσης Place δεν θα είναι ίδιο με κανένα στιγμιότυπο της κλάσης Person.

Στιγμιότυπα

owl:AllDifferent - Το στοιχείο που περιέχει τα διακριτά στιγμιότυπα.

owl:distinctMembers- Περιέχει μία λίστα με διακριτά στιγμιότυπα. Χρησιμοποιείται αμέσως μετά την δήλωση του στοιχείου owl:AllDifferent.

owl:differentFrom - Δηλώνει ότι ένα στιγμιότυπό είναι διαφορετικό από ένα άλλο.

Παράδειγμα Υλοποίησης

```
<rdf:Description rdf:ID="Tiger">
<rdf:type rdf:resource="#Animal"/>
</rdf:Description>

<rdf:Description rdf:ID="Elephant">
<rdf:type rdf:resource="#Animal"/>
<owl:differentFrom rdf:resource="Tiger"/>
</rdf:Description>
```

Ορίζουμε δύο στιγμιότυπα τύπου Animal και τα οποία είναι διαφορετικά μεταξύ τους.

Προκαθορισμένες Κλάσεις

owl:Nothing - Κάθε κλάση που δημιουργείται έχει υποκλάση την Nothing.

owl:Thing - Κάθε κλάση που δημιουργείται έχει υπερκλάση της Thing.

Πίνακας 2.8 – Στοιχεία OWL

Ολοκληρώνοντας, η τρέχουσα έκδοση της γλώσσας OWL είναι η OWL2, όπως αυτή ανακοινώθηκε και καθιερώθηκε από την W3C, στις 27 Οκτωβρίου 2009. Η νέα έκδοση παρέχει περισσότερο εκφραστική ικανότητα καθώς κάποιες διευκολύνσεις στην δήλωση των στοιχείων του λεξιλογίου της OWL.

2.5 Ασαφεια

Όπως αναφέραμε και στο κεφάλαιο 1, ένας από τους στόχους της παρούσας διπλωματικής εργασίας, είναι η υλοποίησης μας διαδικτυακής πλατφόρμας (wbGraphFuzzyOnto) με απώτερο στόχο την καλύτερη διαχείριση και αξιοποίηση των

οντολογιών, αντιμετωπίζοντας το πρόβλημα της Ασάφειας. Για αυτό τον λόγο, σε αυτό το κεφάλαιο θα αναφερθούμε στο πρόβλημα της Ασάφειας, εστιάζοντας σε κομμάτια της Ασαφούς Λογικής, όπως είναι το Ασαφές Σύνολο, που σχετίζονται άμεσα με την αναπαράσταση και έκφραση ασαφών δεδομένων με σημασιολογικούς όρους. Επίσης μέσα από την περιγραφή και από τα παραδείγματα που παρουσιάζονται, δίνεται η ευκαιρία στον αναγνώστη να κατανοήσει επακριβώς τον ορό της «Ασαφοποίησης».

Η βασική αρχή της Ασαφούς Λογικής (Fuzzy Logic) είναι ότι μπορούμε να ορίσουμε μία πρόταση ως αληθής εκφράζοντας την βάση του βαθμού βεβαιότητας - αλήθειας που εμπεριέχει η πρόταση και όχι χαρακτηρίζοντας την με ακριβή όρο αληθής ή ψευδής. Για παράδειγμα, αν είχαμε την εξής πρόταση: Ο άνθρωπος που έχει ύψος μεγαλύτερο από 1.80 εκατοστά είναι ψηλός. Βάση της Ασαφούς Λογικής δεν σημαίνει ότι η πρόταση «ο άνθρωπος που έχει ύψος 1.75 εκατοστά είναι ψήλος » είναι ψευδής αλλά είναι αληθής με έναν βαθμός αλήθειας π.χ. 90%.

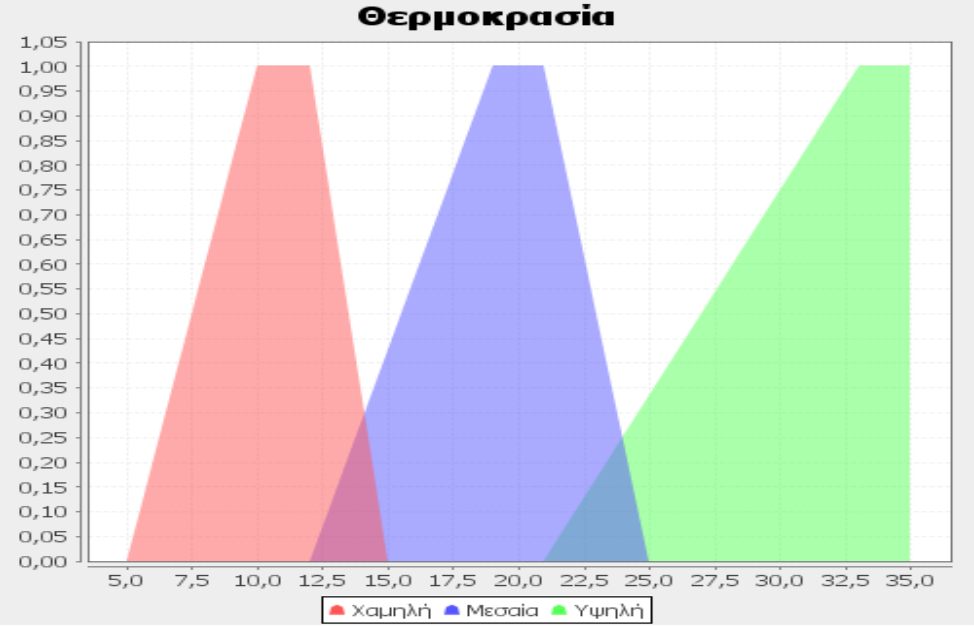
2.5.1 *ΘΕΩΡΙΑ ΑΣΑΦΩΝ ΣΥΝΟΛΩΝ*

Το ασαφές σύνολο αποτελεί ίσως την βασικότερη έννοια του οικοδομήματος της Ασαφούς Λογικής. Η θεωρία των Ασαφών Συνόλων ήρθε να δώσει λύση σε μία λανθασμένη πεποίθηση που αναπτύχθηκε στον χώρο της επιστήμης, ιδιαίτερα μετά τον Β' Παγκόσμιο Πόλεμο και σε συνδυασμό με την αλματώδη ανάπτυξη των υπολογιστών, ότι η τεχνολογική πρόοδος θα εξασφάλιζε την επίλυση όλο και πιο υπολογιστικά απαιτητικών προβλημάτων στο μέλλον.

Ο Lotfi A.Zadeh[10] παρατήρησε ότι ο παραδοσιακός τρόπος περιγραφής ενός συστήματος που στηρίζεται στην αυστηρή μαθηματική λογική (μία μεταβλητή που αντιστοιχεί σε μία πληροφορία είτε ανήκει είτε δεν ανήκει σε ένα υποσύνολο του πεδίου ορισμού της), επηρεάζεται άμεσα από την αύξηση της πολυπλοκότητας ενός συστήματος, με αποτέλεσμα να έχουμε απώλεια πληροφορίας. Στην συνέχεια ο Zadeh, διαδραμάτισε σημαντικό ρόλο στην επίλυση του προβλήματος αυτού, με την δημοσίευση του το 1965 σχετικά με τα ασαφή σύνολα. Συγκεκριμένα τα ασαφή σύνολα σε αντίθεση με τα παραδοσιακά της κλασικής συνολοθεωρίας, διακρίνονται από το γεγονός ότι δεν διαθέτουν αυστηρά όρια με αποτέλεσμα τα αντικείμενα του κόσμου να εντάσσονται ή όχι στο εσωτερικό τους κατά ένα βαθμό συμμετοχής και όχι απόλυτα. Ο βαθμός αυτό κυμαίνεται στο κλειστό διάστημα [0,1], με το 0 να καταδεικνύει ότι μια μεταβλητή που αντιστοιχεί σε μια πληροφορία δεν ανήκει σε καμία περίπτωση στο εξεταζόμενο σύνολο σε αντίθεση με την τιμή 1 που σημαίνει ότι αποτελεί σε κάθε περίπτωση μέλος του συνόλου.

Στο σημείο αυτό παραθέτουμε ένα παράδειγμα. Ας θεωρήσουμε μία μεταβλητή που αντιστοιχεί στην θερμοκρασία ενός συγκεκριμένου νομού μιας χώρας π.χ. νόμος Αττικής και θα έπρεπε να απαντήσουμε στην ερώτηση «Πότε θα λέγαμε ότι μία θερμοκρασία χαρακτηρίζεται μεσαία;». Μία απάντηση μας θα μπορούσε να ήταν όταν η θερμοκρασία για το συγκεκριμένο νομό είναι 20 °C. Αυτό όμως σημαίνει ότι η θερμοκρασία 18 °C δεν μπορεί να χαρακτηριστεί μεσαία παρόλο που έχει πολύ μικρή διαφορά. Η παραπάνω μοντελοποίηση μόνο λανθασμένη θα μπορούσε να χαρακτηριστεί.

Κάνοντας χρήση των ασαφών συνόλων, θα μπορούσαμε να παραστήσουμε γραφικά (εικόνα 2.8) το παραπάνω παράδειγμα, συνάρτηση της βαθμού συμμετοχής (με τιμή από 0 έως 1) δηλαδή ότι μια συγκεκριμένη τιμής της θερμοκρασίας ανήκει σε μία συγκεκριμένη έννοια (Χαμηλή, Μεσαία ή Υψηλή Θερμοκρασία), οπού κάθε έννοια αντιστοιχεί σε ένα υποσύνολο τιμών και ουσιαστικά αποτελεί ένα ασαφές σύνολο. Στον οριζόντιο άξονα δίνεται η τιμής της θερμοκρασίας και στον κάθετο η τιμή της εκάστοτε συνάρτησης συμμετοχής. Έτσι, βάση του διαγράμματος θα μπορούσαμε να συμπεράνουμε ότι η θερμοκρασία 12,5 °C θα μπορούσε να εκφραστεί ως μεσαία με βαθμό συμμετοχής 0,06 και ως χαμηλή με βαθμό συμμετοχής 1.



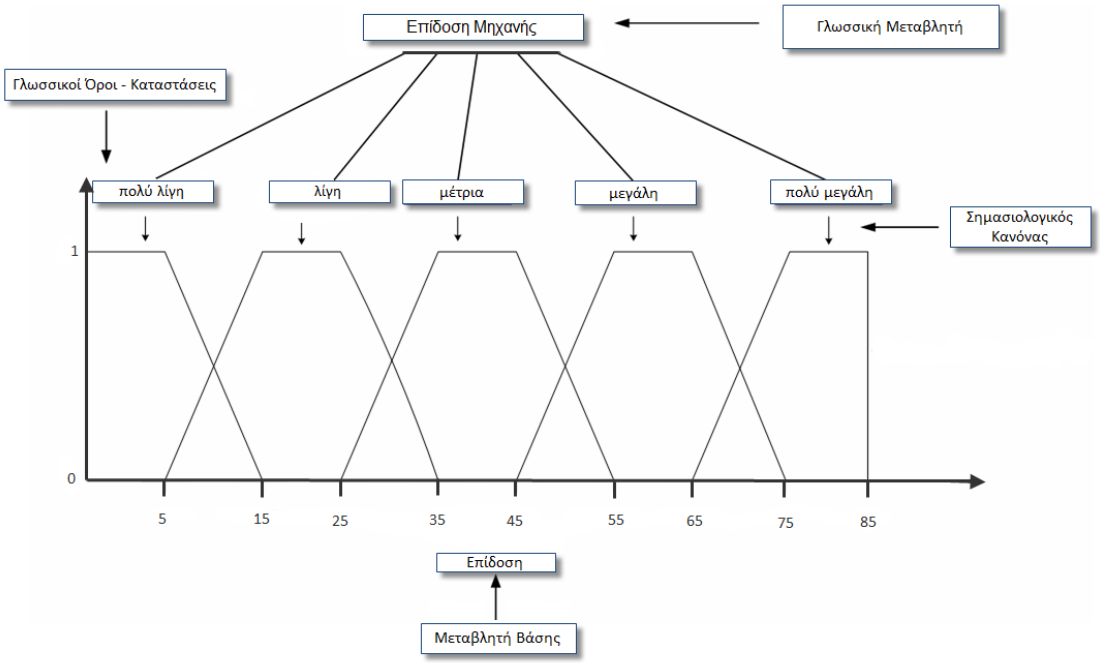
Εικόνα 2.8 – Γραφική παράσταση Ασαφών Συνόλων

Οι έννοιες «Χαμηλή», «Μεσαία» και «Υψηλή» έχουν χρησιμοποιηθεί για την λεκτική αναπαράσταση του ορού θερμοκρασία. Σημαντικό στο σημείο αυτό είναι να αναφέρουμε, ότι ο βαθμός συμμετοχής μιας συγκεκριμένη τιμής είναι άρρηκτα συνδεδεμένος με το περιβάλλον στα πλαίσια του οποίου διατυπώνεται. Για παράδειγμα η τιμής της θερμοκρασίας 12,5 °C θα είχε αποδοθεί διαφορετικά για ένα άλλο νομό.

Στο παραπάνω παράδειγμα ο όρος η θερμοκρασία ενός νομού αποτελεί μια γλωσσική μεταβλητή. Μία γλωσσική μεταβλητή διαθέτει μία μεταβλητή βάσης, οι τιμές της οποίας είναι πραγματικοί αριθμοί σε ένα προκαθορισμένο εύρος. Μία μεταβλητή βάσης πρόκειται ουσιαστικά για ένα μέγεθος του οποίου οι τιμές κυμαίνονται και το οποίο μπορεί να έχει είτε φυσική υπόσταση (όπως στο παράδειγμα μας η θερμοκρασία) είτε να πρόκειται για κάτι μετρήσιμο (π.χ. αξιοπιστία). Στα πλαίσια μιας γλωσσικής μεταβλητής χρησιμοποιούνται διαβαθμίσεις (όπως στο παράδειγμα μας οι έννοιες χαμηλή , μεσαία, υψηλή) οι οποίες αντιστοιχούν σε ασαφή σύνολα.

Αποδίδοντας έναν πιο φορμαλιστικό όρο, μία γλωσσική μεταβλητή αποτελείται από μία πεντάδα (υ, T, X, g, m), όπου υ είναι το όνομα της μεταβλητής, T είναι το σύνολο των

γλωσσικών όρων που αναφέρονται στη μεταβλητή υ και των οποίων η σημασία καλύπτει όλο το εύρος του καθολικού συνόλου X, g είναι ένας γραμματικός κανόνας για την παραγωγή γλωσσικών όρων και m είναι ένας σημασιολογικός κανόνας ο οποίος αναθέτει σε κάθε γλωσσικό όρο t ∈T μία σημασία m(t) η οποία με τη σειρά της αντιστοιχεί σε ένα ασαφές σύνολο στο X. Στην εικόνα 2.9, παρουσιάζουμε το διάγραμμα με τα ασαφή σύνολα που αντιστοιχούν στην γλωσσική μεταβλητή «επίδοση μηχανής».



Εικόνα 2.9 – Διάγραμμα γλωσσικής μεταβλητής «Απόδοσης Μηχανής»

Ο βαθμός συμμετοχής μιας τιμής x σε ένα ασαφές σύνολο συμβολίζεται με «μ(x)». Η περιγραφή μιας μεταβλητής x με λεκτικούς όρους ονομάζεται «διαμερισμός της μεταβλητής» και η απόδοση μιας αυστηρά αριθμητής τιμής με λεκτικούς όρους, όπως για παράδειγμα η τιμής της θερμοκρασίας 12,5 °C ονομάζεται «Ασαφοποίηση»(Fuzzyfication) της «crisp» τιμής.

Από τα διαγράμματα της εικόνας 2.8 και 2.9, μπορούμε επίσης να διακρίνουμε ότι η αβεβαιότητα λαμβάνει μέγιστή τιμή στα όρια, πράγμα λογικό αν αναλογιστεί κανείς ότι αυτά ακριβώς είναι τα σημεία για το οποία δεν μπορούμε να απαντήσουμε με σαφήνεια σε πια κατάσταση ακριβώς βρισκόμαστε και για αυτό τον λόγο τα ασαφή σύνολα σε αυτές τις περιπτώσεις μας δίνουν πιο ακριβή αποτελέσματα. Είναι σημαντικό να επισημάνουμε ότι η ασαφής λογική δεν θα πρέπει να συνδέεται με την στατιστική ανάλυση διότι ένα ασαφές σύνολο εκφράζει κατανομή δυνατότητας και ένας βαθμός συμμετοχής μιας τιμής για μία μεταβλητή αποτελεί τον βαθμό βεβαιότητας ότι αυτό που διατυπώνουμε είναι αληθές και όχι ότι είναι πιθανόν να συμβεί βάση κάποια συγκεκριμένης πιθανότητας.

Στα διαγράμματα της εικόνας 2.8 και 2.9 έχουμε χρησιμοποιήσει για να παραστήσουμε την μορφή των βαθμών συμμετοχής τραπεζοειδές σχήμα αλλά αυτό δεν είναι απαραίτητο, ούτε υποχρεωτικό, μας δίνεται η δυνατότητα να χρησιμοποιήσουμε ακόμη τριγωνοειδές ή γκαουσιανό σχήμα.

Εικόνα 2.10 – Μορφές Απεικόνισης Βαθμών Συμμετοχής

Προχωράμε σε μια σειρά από ορισμούς για τα ασαφή σύνολα. Όταν ένα ασαφές σύνολο έχει πεδίο ορισμού X που αποτελείται από διακριτές και πεπερασμένες τιμές (x1, x2 έως xn), τότε το ασαφές σύνολο αναπαριστάται με την ένωση των διαταγμένων ζευγών $x_i/\mu(x_i)$ με i:1 έως n, $x_i \in X$ και με $\mu(x_i) \in [0,1]$, όπου $\mu(x_i)$ ο αντίστοιχος βαθμός συμμετοχής της τιμής x_i. A-cut ενός ασαφούς συνόλου A με συνάρτηση συμμετοχής $\mu_A(x)$ είναι ένα νέο ασαφές σύνολο A' με συνάρτηση συμμετοχής:

$$\mu_\alpha(x) = \begin{cases} \mu_A(x) \text{ εάν } 0 \leq \mu_A(x) \leq \alpha \\ \alpha \text{ εάν } \alpha \leq \mu_A(x) \leq 1 \end{cases}$$

όπου αν το α ισούται με 1 τότε το νέο ασαφές σύνολο που προκύπτει ισοδυναμεί με το αρχικό. Στήριγμα ενός ασαφούς συνόλου A ονομάζεται το κλασικό εκείνο σύνολο το οποίο περιλαμβάνει όλα εκείνα τα στοιχεία του καθολικού συνόλου X τα οποία λαμβάνουν μη μηδενική τιμή στη συνάρτηση συμμετοχής μ. Επιπλέον πυρήνα ενός ασαφούς συνόλου A ονομάζουμε το σύνολο εκείνο στο οποίο ανήκουν όλα τα στοιχεία του καθολικού συνόλου X των οποίων η τιμή της συνάρτησης συμμετοχής μ είναι ίση με 1. Μια εναλλακτική διατύπωση του πυρήνα ενός ασαφούς συνόλου A είναι το 1-cut του A. Ύψος ενός ασαφούς συνόλου A ονομάζουμε τη μέγιστη τιμή που λαμβάνει η συνάρτηση συμμετοχής μ και το συμβολίζουμε με h(A). Εάν ισχύει h(A)=1 τότε το ασαφές σύνολο ονομάζεται κανονικό ενώ εάν έχουμε h(A)<1 τότε το ασαφές σύνολο ονομάζεται υπο-κανονικό.

Όπως στα κλασικά σύνολα ορίζουμε μια σειρά από πράξεις μεταξύ συνόλων, παρομοίως και στα ασαφή σύνολα μπορούμε να ορίσουμε μία σειρά από πράξεις. Οι βασικότερες πράξεις που ορίζονται μεταξύ ασαφών συνόλων είναι ακόλουθες:

- Η Ένωση (Union)
- Η Τομή (Intersection)
- Το Συμπλήρωμα (Complement) ενός ασαφούς συνόλου

Συγκεκριμένα, αν υποθέσουμε ότι έχουμε δύο ασαφή σύνολα A και B που ορίζονται στο πεδίο ορισμού X, τότε η ένωση μεταξύ των συνόλων A και B δίνεται από την σχέση:

- $\mu_{A \cup B}(x) = \mu_A(x) \cup \mu_B(x) \ \forall x \in X,$

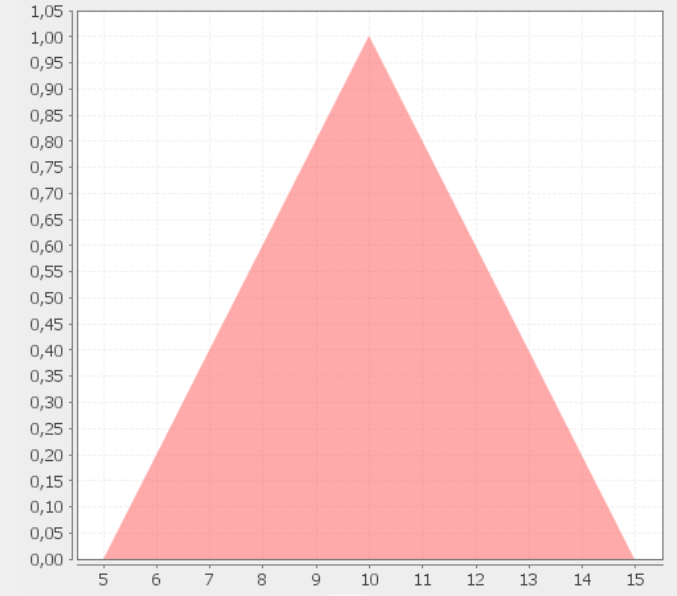
Η τομή των ασαφών συνόλων A και B από την σχέση:

- $\mu_{A\cap B}(x)= \mu_A(x)\cap\mu_B(x)\ \forall x\in X,$

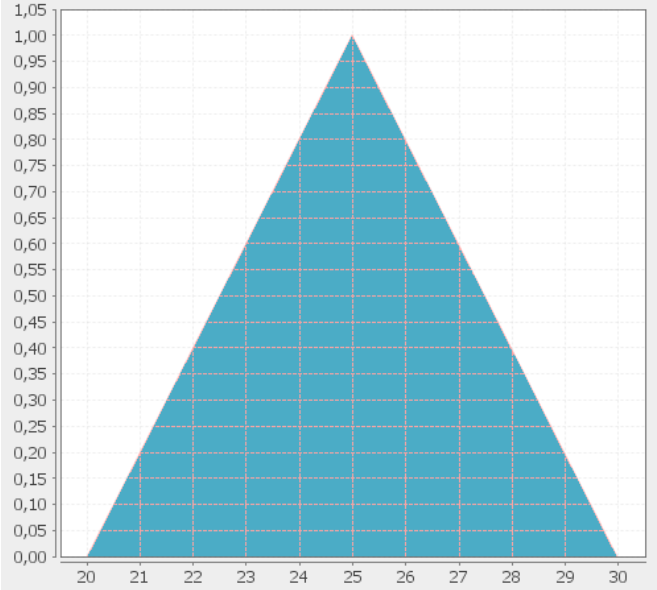
ενώ το συμπλήρωμα ενός ασαφούς συνόλου Α από την σχέση :

- $\mu_{\neg A}(x)= 1-\mu_A(x)\ \ \forall x\in X.$

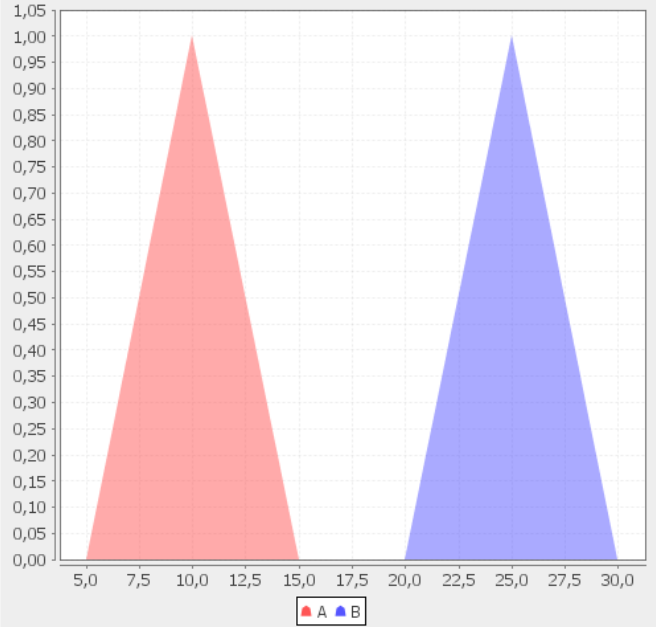
Επίσης για να δηλώσουμε ότι ένα ασαφές σύνολο είναι υποσύνολο κάποιου άλλου ασαφούς συνόλου, αυτό το ορίζουμε βάση της συνάρτησης συμμέτοχης τους. Για παράδειγμα, δεδομένου δυο ασαφών συνόλων Α και Β για να δηλώσουμε ότι το Α είναι υποσύνολο του Β ($A\subset B$), θα πρέπει να ισχύει $A(x)\leq B(x)$ για κάθε x που ανήκει στο πεδίο ορισμού Χ, όπου Α(x) και Β(x) οι συναρτήσεις συμμετοχής των ασαφών συνόλων Α και Β αντίστοιχα. Στα επόμενα σχήματα που ακλουθούν απεικονίζουμε τα αποτελέσματα των παραπάνω πράξεων, Ένωσης, Τομής και Συμπληρώματος δύο ασαφών συνόλων Α και Β με σκοπό την καλύτερη κατανόηση τους.



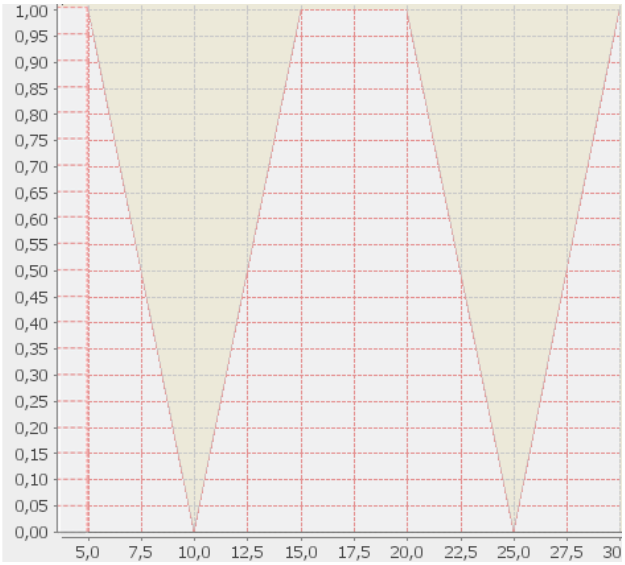
Εικόνα 2.11 – Ασαφές σύνολο Α



Εικόνα 2.12 - Ασαφές σύνολο B



Εικόνα 2.13 – Ασαφής ένωση των A και B



Εικόνα 2.14 – Ασαφές συμπλήρωμα της ασαφούς ένωσης των A και B

2.5.2 *ΑΣΑΦΕΙΣ ΟΝΤΟΛΟΓΙΕΣ*

Όπως έχουμε ήδη αναφέρει, τις τελευταίες δεκαετίες καταβάλλεται μεγάλη προσπάθεια από τους επιστήμονές για την μελέτη της Αβεβαιότητας δηλαδή της έλλειψης ακρίβειας και της ύπαρξη ασάφειας στην αναπαράσταση γνώσης και κατά συνέπεια και στις οντολογίες. Μία κατεύθυνση που ακολουθήθηκε για την αντιμετώπιση της ασάφειας κατά την δημιουργία μια οντολογίας, από την έλλειψη απόδοσης με ακριβείς σημασιολογικούς όρους ασαφών δεδομένων, είναι η ασαφής επέκταση της σημασιολογικής γλώσσας OWL. Το αποτέλεσμα της προσπάθεια για την ασαφή επέκταση της γλώσσα OWL είναι η γλώσσα f-OWL.

2.5.3 *F-OWL*

Για την πλειοψηφία των εφαρμογών του σημασιολογικού ιστού είναι αναγκαία η δυνατότητα διαχείρισης της ασάφειας και της αβεβαιότητας κατά την περιγραφή - αναπαράσταση πληροφοριών και σε πολλές περιπτώσεις χρησιμότερη από αυτήν της δημιουργίας σχέσεων μεταξύ κλάσεων (δηλαδή εννοιών). Σε αυτό το πλαίσιο καθορίστηκε η ασαφής επέκταση της OWL, η f-OWL[11], η οποία έχει ως κύρια διαφοροποίηση την ύπαρξη βαθμών συμμετοχής στους ισχυρισμούς (γεγονότα).

Η επέκταση της f-OWL απαιτεί και καθορισμό νέας σύνταξης, τόσο σε αφηρημένο επίπεδο όσο και σε επίπεδο σύνταξης RDF/XML, έτσι ώστε να είναι δυνατή, η δήλωση του βαθμού συμμετοχής. Ορίζεται λοιπόν το στοιχείο «membership», το οποίο αποτελείται από δύο επιμέρους στοιχεία, τα «ineqType» και «degree». Το «ineqType» δηλώνει το είδος της ανισότητας (π.χ. >, <), ενώ το «degree» υποδηλώνει το βαθμό συμμετοχής του ατόμου(στιγμιότυπο μια κλάσης) στη συγκεκριμένη κλάση και έχει πεδίο τιμών το [0, 1](όπως και στα ασαφή σύνολα). Στην περίπτωση που παραλείπεται ο βαθμός, υπονοείται

ότι ο τύπος ανισότητας ισοδυναμεί με « = » και βαθμός συμμετοχής ίσος με «1». Όσον αφορά την σύνταξη της f-OWL, δίνεται παρακάτω στον πίνακα 2.9, ένα παράδειγμα υλοποίησης οντολογίας με f-OWL, περιγραφής στοιχείων ενός ατόμου, όπου έχουμε ασαφοποίηση της ιδιότητας «ψηλός»(tall) και της σχέσης του με κάποιο άλλο άτομο όσον αφορά το βάρος του «heavier».

```
<owl :Description rdf : about="John">
        <owl : type rdf : resource="tall" owlx:ineqType="≥"
                owlx:degree="0.8"/>

                <heavier rdf:resource="Costas" owlx:ineqType="≥"
                owlx:degree="0.5"/>
</ owl :Description>
```

Δηλώνουμε ότι ο John είναι ψηλός σε βαθμό τουλάχιστον 0.8.

Δηλώνουμε ότι ο John είναι πιο βάρης από τον Costa σε βαθμό τουλάχιστον 0.5.

Πίνακας 2.9 – Παράδειγμα Υλοποίησης f-OWL

3 ΜΕΘΟΔΟΛΟΓΙΕΣ ΠΕΡΙΓΡΑΦΗΣ ΟΝΤΟΛΟΓΙΩΝ

Με τα όσα έχουν παρουσιαστεί μέχρι τώρα στον αναγνώστη, γίνεται αντιληπτό ότι δεν έχει υιοθετηθεί από του επιστήμονες κάποιος συγκεκριμένος τρόπος για την ανάπτυξη οντολογιών. Στο κεφάλαιο αυτό θα αναφερθούμε στα μοντέλα ανάπτυξης και στις κυριότερες μεθοδολογίες περιγραφής οντολογιών.

Πριν ξεκινήσουμε την αναφορά μας, επισημάνουμε παρακάτω κάποιες βασικές αρχές σχετικά με την περιγραφή οντολογιών:

- Δεν υπάρχει κανένας σωστός τρόπος για την περιγραφή ενός γνωστικού πεδίου. Πάντα υπάρχουν συνήθως πολλοί εναλλακτικοί τρόποι για την περιγραφή μιας γνωστικής περιοχής. Ο καλύτερος τρόπος είναι άρρηκτα συνδεδεμένος με την εφαρμογή που έχουμε υπόψη καθώς και με τις πιθανές μελλοντικές επεκτάσεις της.

- Οι έννοιες που υιοθετούνται σε μία οντολογία πρέπει να είναι κοντά στα αντικείμενα (φυσικά ή λογικά) και οι σχέσεις στο πεδίο ενδιαφέροντος της συγκεκριμένης γνωστικής περιοχής. Σε μία πρόταση μία οντολογίας χρησιμοποιούμε συνήθως ρήματα για την περιγραφή των σχέσεων.

- Η δημιουργία μιας οντολογίας είναι ουσιαστικά μια επαναληπτική διαδικασία.

Μία οντολογία αποτελεί μία αναπαράσταση γνώσης συνόλων του πραγματικού κόσμου και οι έννοιες που απαρτίζουν μία οντολογία θα πρέπει να αποδίδουν την γνώση αυτή με έγκυρο τρόπο. Για αυτό τον λόγο, μετά τον καθορισμό μιας αρχικής έκδοσης μια οντολογίας μπορούμε να την αξιολογήσουμε και να την διορθώσουμε με τη χρησιμοποίηση της στις εφαρμογές ή με μεθόδους επίλυσης προβλημάτων ή με άτυπες συζητήσεις με τους εμπειρογνώμονες της συγκεκριμένης γνωστικής περιοχής. Αυτή η διαδικασία αξιολόγησης και διόρθωσης συνεχίζεται σε ολόκληρο τον κύκλο ζωής της οντολογίας.

Υπάρχουν τρία βασικά μοντέλα ανάπτυξης οντολογιών[17]. Η top-down (από-πάνω προς-τα-κάτω) ανάπτυξη, bottom-up (από-κάτω-προς-τα-πάνω) ανάπτυξη και η middle-out (ενδιάμεση) ανάπτυξη.

Συμφώνα με το μοντέλο **top - down**, αρχικά αναπτύσσουμε μια οντολογία με την οποία αναπαριστούμε την γνωστική περιοχή, χρησιμοποιώντας θεμελιώδεις έννοιες του πεδίου. Κατόπιν επεκτείνουμε και εμπλουτίζουμε την βασική αυτή οντολογία με περισσότερες έννοιες έως ότου φθάσουμε στο σημείο η οντολογία μας να περιγράφει όλα τα δεδομένα που έχουν προσδιοριστεί. Βασική απαίτηση αυτή της προσέγγισης είναι ο προσδιορισμός των βασικών εννοιών της γνωστικής περιοχής καθώς και της συσχέτισή τους.

Αντίθετα με το top – down μοντέλο, στο bottom - up μοντέλο χρησιμοποιούμε ως σημείο εκκίνησης τα δεδομένα. Αναπτύσσουμε οντολογίες που περιγράφουν συγκεκριμένα

δεδομένα με μεγάλη ακρίβεια αφού στηρίζονται άμεσα σε αυτά. Στην συνέχεια με την διασύνδεση και την επέκταση των οντολογιών αυτών επιτυγχάνουμε όλο και μεγαλύτερου επιπέδου αφαίρεση και καταλήγουμε σε μια ολοκληρωμένη οντολογία που περιγράφει μια συγκεκριμένη γνωστική περιοχή. Μειονέκτημα αυτού του μοντέλου είναι ότι χτίζοντας την οντολογία ξεκινώντας από τα δεδομένα, στηριζόμαστε στις ιδιαιτερότητες αυτών, με αποτέλεσμα η τελική οντολογία να μην παρέχει πλήρη και ακριβή περιγραφή της γνωστικής περιοχής.

Το μοντέλο **middle - out** αποτελεί έναν συνδυασμό των παραπάνω μοντέλων. Βασίζεται στις βασικές οντότητες της οντολογίας δημιουργώντας κανόνες και σχέσεις μεταξύ τους, προχωρώντας αφαιρετικά σε πιο υψηλά επίπεδα σχεδίασης. Και για τα τρία μοντέλα ανάπτυξης οντολογιών, δίνεται η δυνατότητα στο χρήστη να χρησιμοποιήσει και ενσωματώσει ήδη υπάρχουσες υλοποιημένες οντολογίες.

Πάνω σε αυτά τα βασικά μοντέλα ανάπτυξης, στηρίχθηκαν οι προσπάθειες για την ανάπτυξη συγκεκριμένων μεθοδολογιών περιγραφής οντολογιών και το αποτέλεσμα αυτών των προσπαθειών περιγράφουμε στην συνέχεια.

3.1 Μεθοδολογια Uschold και King's

Η μεθοδολογία αυτή βασίστηκε στην εμπειρία που αποκτήθηκε κατά την ανάπτυξη μιας οντολογίας με το όνομα Enterprise Ontology η οποία αφορά την μοντελοποίηση διαφόρων διαδικασιών σε μια επιχείρηση. Η συγκεκριμένη μεθοδολογία αποτελείται από τα παρακάτω βήματα με την ακόλουθη σειρά εκτέλεσης [18]:

1. **Αναγνώριση του σκοπού (Purpose Identification).** Προσδιορισμός του λόγου για τον οποίο κατασκευάζεται η οντολογία, καθώς και περιγραφή των τελικών χρηστών της εφαρμογής.
2. **Χτίσιμο της οντολογίας(Ontology Build).**
   - **2.1.** Σύλληψη (capture) της οντολογίας:
     - **2.1.1.** Ορισμός πεδίου δράσης, αναγνώριση δηλαδή, των σημαντικών εννοιών και των σχέσεων τους για την γνωστική περιοχή που επιθυμούμε.
     - **2.1.2.** Παραγωγή ορισμών για τις έννοιες και τις σχέσεις τους, που να είναι ακριβείς και σαφείς.
     - **2.1.3.** Αναγνώριση όρων που αναφέρονται στις έννοιες και στις σχέσεις τους.
     - **2.1.4.** Συμφωνία όλων των παραπάνω βημάτων.
   - **2.2.** Κωδικοποίηση (Coding). Αναπαράσταση της γνώσης του βήματος 2.1 με τη βοήθεια μιας τυπικής σημασιολογικής γλώσσας, όπως οι γλώσσες που αναφέρθηκαν στο κεφάλαιο 2.

2.3. Ενοποίηση(Integration) υπαρχόντων οντολογιών*:* Στα βήματα της σύλληψης και της κωδικοποίησης υπάρχει πάντα η ανησυχία για το αν και πώς, μπορούν να χρησιμοποιηθούν ήδη υπάρχουσες οντολογίες. Αυτό ουσιαστικά πρόκειται για ένα πολύ δύσκολο επιχείρημα το οποίο για να το επιτύχουμε θα πρέπει να έχει γίνει σαφές ο ακριβής ρόλος κάθε οντολογίας.

3. **Αξιολόγηση(Evaluation):** Στο στάδιο αυτό αξιολογείται η οντολογία, το περιβάλλον ανάπτυξης αυτής και τα κείμενα τεκμηρίωσής της σε σχέση με ένα πλαίσιο αναφοράς. Πλαίσιο αναφοράς μπορεί να χαρακτηριστεί το έγραφα απαιτήσεων (requirements specification), ερωτήσεις ικανότητας - επάρκειας (competency questions) ή και ο πραγματικός κόσμος.
4. **Τεκμηρίωση(Documentation):** Τελική τεκμηρίωση της οντολογίας βάση της αναφοράς που έχει πραγματοποιηθεί στο βήμα 1.

3.2 Μεθοδολογια ON-TO-KNOWLEDGE

Η μεθοδολογία On-To-Knowledge[19] αναπτύχθηκε και εφαρμόστηκε στο ευρωπαϊκό έργο Eu IST-199910132. Σκοπός της είναι η βελτίωση της ποιότητας των επιχειρηματικών εφαρμογών διαχείρισης γνώσης, όπως για παράδειγμα μία εφαρμογή διαχείρισης ανθρωπινού δυναμικού μιας επιχείρησης, με την χρήση οντολογιών για την αναπαράσταση της πληροφορίας. Η μεθοδολογία καλύπτει τόσο τις αρχικές φάσεις ανάπτυξης ενός έργου διαχείρισης γνώσης όσο και την τελική φάση δημιουργίας ενός συστήματος διαχείρισης γνώσης βασισμένο σε οντολογίες. Υπάρχουν διάφορες μεθοδολογίες που υποστηρίζουν τη συστηματική εύρεση λύσεων διαχείρισης γνώσης μέσα σε επιχειρήσεις. Η CommonKADS μεθοδολογία αποτελεί χαρακτηριστικό παράδειγμα η οποία δίνει έμφαση σε μία πρώιμη μελέτη σκοπιμότητας (feasibility study) όπως επίσης και στη κατασκευή πολλών μοντέλων που καταγράφουν διαφορετικού τύπου γνώση που χρειάζεται για την δημιουργία μίας λύσης διαχείρισης γνώσης. Όπως παρουσιάζεται και στην εικόνα 3.1, η μεθοδολογία On-To-Knowledge περιλαμβάνει πέντε βασικά βήματα:

- Μελέτη Σκοπιμότητας (Feasibility Study). Αυτό το βήμα λαμβάνει χώρα πριν την έναρξη ανάπτυξης της οντολογίας και εξετάζει το κατά πόσο είναι δυνατή η ανάπτυξη της οντολογίας που πρόκειται να αναπτυχθεί. Αναγνωρίζονται τα προβλήματα που τυχόν μπορεί να προκύψουν καθώς επίσης διατυπώνονται και οι πιθανές λύσεις τους. Η μελέτη σκοπιμότητας συντελεί στην λήψη αποφάσεων σχετικά με τις οικονομικές και τεχνολογικές πτυχές του έργου καθώς επίσης με τον προσδιορισμό του σεναρίου χρήσης της οντολογία και των τελικών χρηστών.

- Εκκίνηση (Kickoff). Σε αυτό το βήμα καθορίζονται οι απαιτήσεις, ο στόχος, οι τελικοί χρήστες καθώς και οι κατευθυντήριες γραμμές σχεδιασμού της οντολογίας. Όλα τα παραπάνω συνθέτουν το έγγραφο προδιαγραφών και απαιτήσεων (requirements specification documents) που συντάσσεται σε αυτό το βήμα. Επίσης, εξετάζεται και το κατά πόσο είναι εφικτό η χρησιμοποίηση

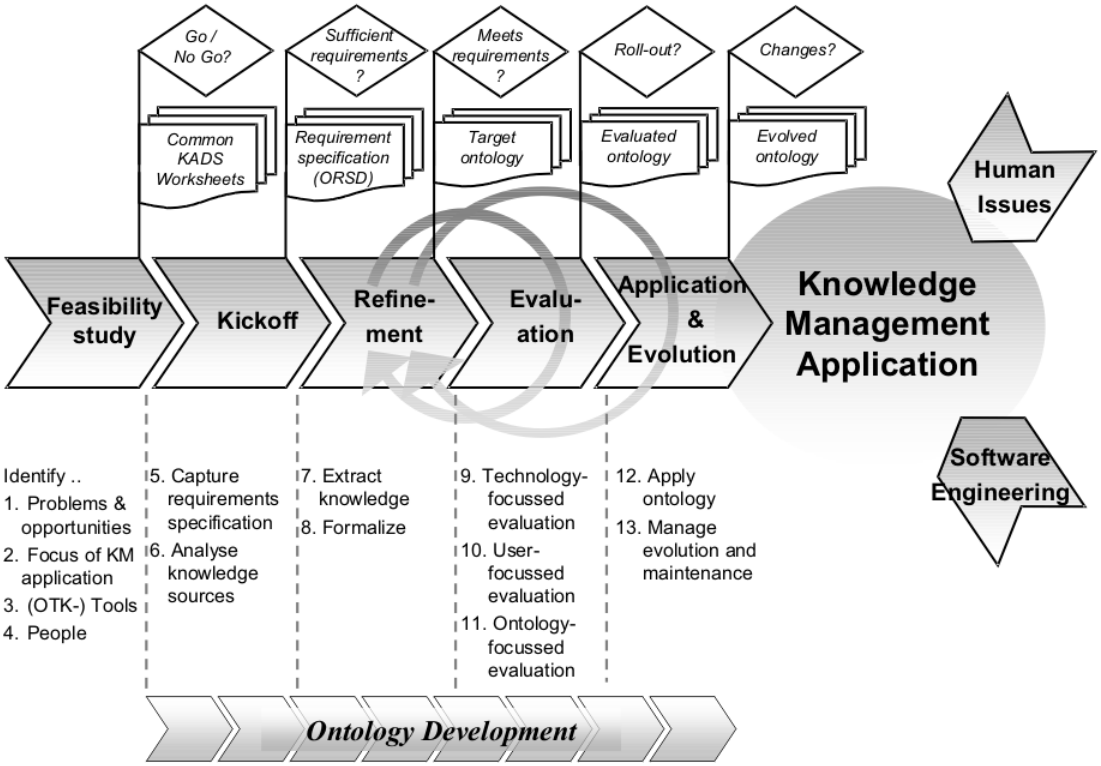
οντολογιών που έχουν ήδη αναπτυχθεί και μπορούν να επαναχρησιμοποιηθούν για την ανάπτυξη της νέας οντολογίας.

- Επεξεργασία (Refinement). Στόχος είναι η παραγωγή μιας λειτουργικής οντολογίας η οποία θα εκπληρώνει τοις προδιαγραφές ορίστηκαν στο προηγούμενο βήμα. Δύο είναι οι ενέργειες που πρέπει να γίνουν σε αυτό το στάδιο:

  - *Εξαγωγή γνώσης* (Extract Knowledge) από τους ειδικούς (domain experts) για το γνωστικό πεδίο στο οποίο θέλουμε να εφαρμοστεί η υπό ανάπτυξη οντολογία.
  - *Τυποποίηση* (formalization) της γνώσης που έχει εξαχθεί στην προηγούμενη ενέργεια. Περιγράφουμε την γνώση που αποκτήσαμε με μία σημασιολογική γλώσσα.

- Αξιολόγηση (Evaluation). Σε αυτό το βήμα πραγματοποιείται η αξιολόγηση της παραχθείσας οντολογίας. Αρχικά γίνεται έλεγχος για το αν η παραχθείσα οντολογία εκπληρώνει τις προδιαγραφές οι οποίες καθορίστηκαν στο δεύτερο βήμα, και στην συνεχεία πραγματοποιείται έλεγχος λειτουργίας της οντολογίας στο περιβάλλον εφαρμογής για το οποίο αναπτύχθηκε.

- Συντήρηση (Maintenance). Επειδή μία οντολογία αποτελεί αναπαράσταση γνώσης που προέρχεται από τον πραγματικό κόσμο, συνεπάγεται από αυτό η απαίτηση για την παροχή δυνατότητας αλλαγής και τροποποίησης της οντολογίας. Σε αυτό το βήμα προβλέπεται και σχεδιάζεται η συντήρηση της οντολογίας.



Εικόνα 3.1 - Τα στάδια της μεθοδολογίας On-To-Knowledge

3.3 ΜΕΘΟΔΟΛΟΓΙΑ METHODOLOGY

Η μεθοδολογία Methodology[20] αναπτύχθηκε στο εργαστήριο Τεχνητής Νοημοσύνης στο πολυτεχνείο της Μαδρίτης. Η Methodology είναι μία μεθοδολογία που είναι εμπνευσμένη από τις βασικές ενέργειες που αφορούν την διαδικασία ανάπτυξης ενός λογισμικού. Η Methodology περιλαμβάνει:

- Την αναγνώριση της διαδικασίας ανάπτυξης της οντολογίας (identification of the ontology development process).
- Τον κύκλο ζωής βασισμένο σε εξελικτικά πρότυπα.
- Συγκεκριμένες τεχνικές για να υλοποιηθεί κάθε εργασία που αναγνωρίστηκε στην διαδικασία ανάπτυξης οντολογίας.

Με τον όρο εξελικτικά πρωτότυπα (Evolving Prototypes) αναφερόμαστε σε πρωτότυπα που δημιουργούνται κατά την διάρκεια της ανάπτυξης ενός λογισμικού με απώτερο στόχο την παρουσίαση της μορφής που έχει το υπό-ανάπτυξη λογισμικό την δεδομένη χρονική στιγμή. Τα εξελικτικά πρωτότυπα δεν αποτελούν ολοκληρωμένη μορφή του λογισμικού αλλά παρουσιάζουν την λειτουργικότητα των ολοκληρωμένων μερών του υπό-ανάπτυξη λογισμικού οπού μία λειτουργικότητα ενός συγκεκριμένου κομματιού του λογισμικού μπορεί να διαφέρει στην ολοκληρωμένη και τελική έκδοση του λογισμικού.

3.3.1 *Διαδικασια Αναπτυξης Οντολογιων*

Η Διαδικασία Ανάπτυξης Οντολογιών αναφέρεται στις ενέργειες που εκτελούνται κατά την δημιουργία μίας οντολογίας. Στόχος της συγκεκριμένης διαδικασίας δεν είναι η εκτέλεση των ενεργειών αυτών με μία καθορισμένη σειρά απλώς η αναγνώριση τους και η καταγραφή τους. Για την περιγραφή των ενεργειών αυτών χρησιμοποιούμε συνήθως ρήματα. Η συγκεκριμένη διαδικασία αποτελείται από τα εξής ενέργειες:

- Πριν ξεκινήσουμε την δημιουργία της οντολογίας θα πρέπει να έχει προηγηθεί ένας σχεδιασμός – προγραμματισμός σχετικά με τις εργασίες που θα πρέπει να γίνουν, τους πόρους που θα πρέπει να δεσμευτούν για την υλοποίηση της οντολογίας και φυσικά το χρονικό πλαίσιο εντός του οποίου θα πρέπει να ολοκληρωθεί κάθε εργασία. Ο δημιουργός της οντολογίας θα πρέπει να γνωρίζει επακριβώς τον σκοπό και τον στόχο της οντολογίας, το αν είναι εφικτή η δημιουργία της οντολογίας, το πεδίο εφαρμογής της, προβλεπόμενες χρήσεις της καθώς και τους τελικούς χρήστες που πρόκειται να την χρησιμοποιήσουν. Ουσιαστικά το σύνολο των παραπάνω ενεργειών αποτελούν το έγγραφο προδιαγραφών-απαιτήσεων.

- Μόλις αποκτηθεί η γνώση σχετικά με το αντικείμενο που θέλουμε να περιγράψουμε, δημιουργούμε ένα άτυπο μοντέλο περιγραφής των εννοιών του, το οποίο αποτελεί την δομή της γνώσης του. Αυτή η ενέργεια ονομάζεται Σύλληψη της γνώσης (Conceptualization).

- Μετατροπή του εννοιολογικού μοντέλου γνώσης σε μία τυπική μορφή, ουσιαστικά δημιουργία της οντολογίας με την περιγραφής της γνώσης με μια σημασιολογική γλώσσα την οποία θα την καθιστά κατανοητή στους υπολογιστές. Αυτή η ενέργεια ονομάζεται Formalization. Θα πρέπει να χρησιμοποιούνται όσο τον δυνατόν πιο επαναχρησιμοποιήσιμα λεξιλόγια για την δημιουργία μιας οντολογίας και εφόσον είναι εφικτό και ήδη υπάρχουσες οντολογίες.

- Πριν μια οντολογία γίνει διαθέσιμη θα πρέπει να πρώτα να πραγματοποιηθεί αξιολόγηση της σχετικά με το γνωστικό αντικείμενο που περιγράφει.

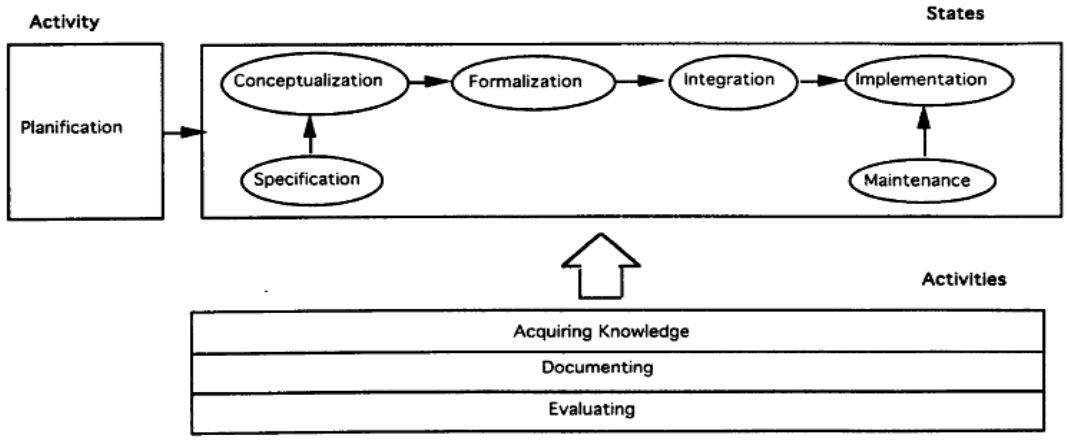
- Εάν εντοπιστεί κάποια έλλειψη στην οντολογία , μπορεί κάποιος να επιστέψει στην φάση ορισμού για να κάνει τις απαιτούμενες αλλαγές. Αυτή η ενέργεια ονομάζεται Συντήρηση(Maintenance).

- Θα πρέπει να υπάρχει ένα έγγραφο τεκμηρίωσης της οντολογίας ώστε να καθίστα την αλλαγή όρων σε μία οντολογία εύκολη διαδικασία αλλά και έγκυρη.

3.3.2 *Κυκλος Ζωης Οντολογιας*

Στην Διαδικασία Ανάπτυξης Οντολογιών ορίσαμε ένα σύνολο ενεργειών χωρίς όμως να ορίσουμε την σειρά με την οποία θα εκτελεστούν καθώς και σε τι βαθμό. Η σειρά αυτή καθορίζεται από τον κύκλο ζωής που προτείνει η Methodology, ο οποίος βασίζεται σε εξελικτικά πρωτότυπα. Για την καλύτερη κατανόηση του αναγνώστη, ο κύκλος ζωής μιας

οντολογίας αποτελεί ένα συγκεκριμένο σύνολο καταστάσεων στις οποίες διέρχεται η οντολογία κατά την δημιουργία της, όπου κάθε κατάσταση αντιστοιχεί σε ένα συγκεκριμένο σύνολο ενεργειών που αντιστοιχούν στις ενέργειες της διαδικασίας ανάπτυξης οντολογιών. Οι καταστάσεις στις οποίες διέρχεται μία οντολογία και με την αντίστοιχή σειρά είναι οι ακόλουθες (εικόνα 3.2) : Προδιαγραφές (Specification), Σύλληψη (Conceptualization), Τυποποίηση (Formalization), Ολοκλήρωση - Ενοποίηση (Integration), Εφαρμογή (Implementation), Συντήρηση (Maintenance).



Εικόνα 3.2 – Κύκλος Ζωής Οντολογίας

Το μεγαλύτερος μέρος της γνώσης, αποκτάται κατά την φάση του ορισμού του έγγραφου προσδιορισμών και απαιτήσεων δηλαδή στην πρώτη κατάσταση του κύκλου ζωής μιας οντολογίας και μειώνεται καθώς η υπό-ανάπτυξη οντολογία διέρχεται από την μία κατάσταση στην άλλη. Για την αποφυγή σφαλμάτων που μπορεί να προκύψουν κατά την δημιουργία μια οντολογίας, αλλά και για την καλύτερη και πιο ορθολογική αντιμετώπιση σφαλμάτων σε περίπτωση που προκύψουν, θα πρέπει να πραγματοποιείται αξιολόγηση και τεκμηρίωση σε κάθε κατάσταση. Σημαντικό στο σημείο αυτό είναι να επισημάνουμε ότι μία υπό-ανάπτυξη οντολογία δεν διέρχεται σε επόμενη κατάσταση αν πρώτα δεν έχουν ολοκληρωθεί όλες οι ενέργειες που θα πρέπει να πραγματοποιηθούν σε κάθε κατάσταση.

3.3.3 *Τεχνικες της Methodology για την Δημιουργια μιας Οντολογιας απο την Αρχη*

Η Methodology προτείνει μία από σειρά από ενέργειες οι οποίες θα πρέπει να περιλαμβάνονται στις καταστάσεις που απαρτίζουν τον κύκλο ζωής μιας υπό-ανάπτυξης οντολογίας. Οι ενέργειες αυτές που προτείνει η Methodology πρόεκυψαν και βασίστηκαν από την γνώση που αποκτήθηκε κατά την δημιουργία της οντολογίας «CHEMICALS». Η συγκεκριμένη οντολογία αναπαριστά γνώση για το πεδίο των χημικών στοιχείων και των κρυσταλλικών δομών. Η Methodology είναι ανεξάρτητη από τις εφαρμογές που θα χρησιμοποιηθεί και το μοντέλο ανάπτυξης που χρησιμοποιεί είναι η «**middle - out**» προσέγγιση.

Η Methodology, όσο αφορά τις ενέργειες που πραγματοποιούνται κατά την δημιουργία του **έγγραφου προδιαγραφών – απαιτήσεων** (Specification) προτείνει να περιλαμβάνονται τα εξής:

- Σκοπός της οντολογίας, προτεινόμενες χρήσεις, έναν συγκεκριμένο σενάριο εφαρμογής της οντολογίας και φυσικά οι τελικοί χρήστες.
- Θα πρέπει εφόσον είναι εφικτό να συμπεριλαμβάνονται όλο των το σύνολο των ορών που θα πρέπει αναπαρασταθούν καθώς και τα χαρακτηριστικά τους. Προτείνεται να δημιουργηθεί ένα λεξιλόγιο κύριων ορών που θα πρέπει να περιλαμβάνει η οντολογία. Επίσης είναι πολύ χρηστικό να ταξινομήσουμε τις κυρίες έννοιες μιας οντολογίας σε δενδρική μορφή. Αυτή η δενδρική αναπαράσταση θα βοηθήσει στην εύρεση των ορών που θα πρέπει να περιλαμβάνει η οντολογία καθώς επίσης και στην εξάλειψη όρων που είναι συνώνυμοι με άλλους όρους ή μη σχετικοί με το γνωστικό πεδίο που θέλουμε να αναπαραστήσουμε.

Επειδή είναι πολύ δύσκολο να χαρακτηριστεί ένα έγγραφο προδιαγραφών και απαιτήσεων πλήρες ως προς τους όρους που περιλαμβάνει, η Methodology προτείνει ότι για να χαρακτηριστεί ένα έγγραφο προδιαγραφών – απαιτήσεων «καλό» και «έγκυρο» θα πρέπει να περιλαμβάνει τις ακόλουθες ιδιότητες:

- Περιεκτικότητα, δηλαδή δεν θα πρέπει να υπάρχουν συνώνυμοι οροί ή οροί μη σχετικοί με το γνωστικό πεδίο.
- Μερική πληρότητα, ως προς τους όρους που θα χρησιμοποιηθούν για την αναπαράσταση της γνώσης της οντολογίας.
- Συνέπεια, σχετικά με τους όρους που απαρτίζουν την οντολογία και της σημασιολογίας τους δηλαδή να μην υπάρχουν αντιθέσεις μεταξύ των σημασιολογικών ερμηνειών.

Η Methodology σχετικά με την διαδικασία **απόκτησης γνώσης** (Knowledge Acquisition) προτείνει τα εξής:

- Άτυπες συνεντεύξεις με τους εμπειρογνώμονες του γνωστικού πεδίου για την δημιουργία ενός προσχέδιου του έγγραφου προδιαγραφών απαιτήσεων.
- Άτυπη ανάλυση των εννοιών μιας οντολογίας που δίνονται από βιβλία και εγχειρίδια σχετικά με το γνωστικό αντικείμενο. Αυτό βοηθάει στην διαδικασία της σύλληψης.
- Επίσημη ανάλυση, έτσι ώστε να περιγράψουμε τις κύριες έννοιες της οντολογίας και τις σχέσεις αλληλεπίδρασης μεταξύ τους.
- Επίσημες συνεντεύξεις με τους εμπειρογνώμονες του γνωστικού πεδίου, για την διευκρίνηση των εννοιών και των σχέσεων της οντολογίας αλλά και αξιολόγηση του εννοιολογικού μοντέλου αφού η διαδικασία της σύλληψης έχει ολοκληρωθεί.

Στην διαδικασία της **σύλληψης** (Conceptualization) δημιουργούμε ένα εννοιολογικό μοντέλο της γνώσης που έχουμε αποκτήσει, έχοντας ως βάση το έγγραφο προδιαγραφών - απαιτήσεων και με στόχο την δημιουργία ενός ολοκληρωμένου λεξιλογίου που θα περιλαμβάνει έννοιες(κλάσεις), σχέσεις και στιγμιότυπα που θα αναπαριστούν την γνώση της οντολογίας μας. Στην συνεχεία μπορούμε να ταξινομήσουμε τις κύριες έννοιες της οντολογίας μας σε ομάδες ανάλογα με την σχέση που έχουν μεταξύ τους. Μόλις ολοκληρώσουμε την ταξινόμηση αναπαριστούμε τις ομάδες σε δενδρική μορφή. Αυτό μας δίνει την δυνατότητα να χωρίσουμε και να αναθέσουμε την διαδικασία ανάπτυξης της οντολογίας σε διαφορετικές ομάδες.

Η Methodology για αυτή διαδικασία προτείνει την δημιουργία ενός εννοιολογικού μοντέλου αναπαράστασης γνώσης που θα απαρτίζεται από καλά ορισμένα στοιχεία, έτσι ώστε να δίνεται η δυνατότητα στους τελικούς χρήστες να αξιολογήσουν και να εξακριβώσουν το πόσο χρήσιμη και χρησιμοποίηση για μια εφαρμογή είναι η υπό-ανάπτυξη οντολογία καθώς επίσης και να προβούν στην σύγκριση με άλλες ήδη υλοποιημένες οντολογίες, που αναφέρονται στο ίδιο γνωστικό αντικείμενο, σχετικά με τον σκοπό της οντολογίας αλλά και την πληρότητα των όρων που περιγράφουν.

Η διαδικασία της **ενοποίησης** (Integration) έχει σαν στόχο την πιο γρήγορη ολοκλήρωση των διαδικασιών δημιουργίας μίας οντολογίας χρησιμοποιώντας λεξιλόγιο από ήδη υλοποιημένες οντολογίες. Η Methodology για την διαδικασία της ενοποίησης (Integration) προτείνει τα ακόλουθα:

- Αναζήτηση και εύρεση υλοποιημένων οντολογιών που αναφέρονται στο ίδιο γνωστικό αντικείμενο και κατά συνέπεια τείνουν να έχουν το ίδιο εννοιολογικό μοντέλο. Αυτό μας εξασφαλίζει ότι οροί που χρησιμοποιούνται στις υλοποιημένες οντολογίες δεν θα έρχονται σε αντίθεση με τους όρους της υπό-ανάπτυξης οντολογίας. Σε περίπτωση που δεν ταιριάζουν οι όροι των ήδη υπαρχουσών οντολογιών με τους όρους της υπό-ανάπτυξη οντολογίας είναι προτιμότερο να ξεκινήσει ο ορισμός της και η υλοποίηση της οντολογίας από την αρχή με μια σημασιολογική γλώσσα.
- Αφού επιλέξουμε τους όρους που επιθυμούμε από άλλες οντολογίες και εφόσον δεν διαταράσσεται η συνέπεια της υπό-ανάπτυξης οντολογίας από την χρησιμοποίηση αυτών των όρων, μπορούμε να προβούμε στην αναζήτηση διαφόρων μηχανισμών μετατροπής των όρων αυτών στην σημασιολογική γλώσσα που επιθυμούμε.

Τέλος η Methodology προτείνει για αυτήν την διαδικασία την δημιουργία ενός έγγραφου, στο οποίο θα αναφέρονται τα ονόματα των οντολογιών που πρόκειται να χρησιμοποιήσουμε, τα ονόματα των όρων που θα χρησιμοποιήσουμε από αυτές τις οντολογίες σε αντιστοιχία με τα ονόματα των ορών που έχουμε ορίσει στην διαδικασία της σύλληψης της οντολογία μας και τα όποια θα περιγραφούν από όρους άλλων οντολογιών.

Στην διαδικασία της **εφαρμογής** (Implementation) θα πρέπει να πραγματοποιήσει η μετατροπή των όρων που επιλεχτήκαν στην φάση της ενοποίησης από άλλες οντολογίες στην σημασιολογική γλώσσα που επιθυμούμε.

Η Methodology προτείνει για αυτήν την φάση της ύπαρξης ενός συντακτικού και λεκτικού «αναλυτή» της σημασιολογικής γλώσσα που επιθυμούμε, για την αποφυγή συντακτικών και λεκτικών σφαλμάτων. Επίσης την ύπαρξη ενός επεξεργαστή κείμενου έτσι ώστε να παρέχεται η δυνατότητα προσθήκης ή αλλαγής κάποιων όρων της οντολογίας. Πολύ χρήσιμο είναι και η ύπαρξη διαφόρων μηχανισμών εύρεσης οντολογιών, εύρεσης όρων που μπορούν να χρησιμοποιηθούν στην υπό-ανάπτυξη οντολογία και μηχανισμών αξιολόγησης της σημασιολογικής γνώσης που αναπαριστά η οντολογία.

Για την διαδικασία **αξιολόγησης** της οντολογίας, η Methodology προτείνει την δημιουργία ενός έγγραφου αξιολόγησης από τους μηχανικούς των που ασχολήθηκαν με την ανάπτυξη της οντολογίας στο οποίο θα πρέπει να αναφέρουν πως πραγματοποιήθηκε η αξιολόγηση της οντολογίας, ποιες τεχνικές αξιολόγησης χρησιμοποιήθηκαν, πια σφάλματα προέκυψαν κατά την ανάπτυξη της οντολογίας αναφέροντας την ενέργεια με την οποία προέκυψε σφάλμα και ποιες πήγες γνώσης χρησιμοποιήθηκαν για την αξιολόγηση.

Μετά την ολοκλήρωση της ανάπτυξης μιας οντολογίας, συνήθως δεν παρέχεται κάποιο έγγραφο τεκμηρίωσης από του μηχανικούς που ασχολήθηκαν με την οντολογία. Όπως έχουμε ήδη αναφέρει αυτό έχει σαν αποτέλεσμα να καθιστά την διαδικασία συντήρησης της οντολογίας αρκετά δύσκολη. Η Methodology προτείνει μετά την ολοκλήρωση κάθε σταδίου του κύκλου ζωής μιας οντολογίας την δημιουργία ενός έγγραφου τεκμηρίωσης για το στάδιο αυτό.

3.4 Μεθοδολογια Diligent

Η μεθοδολογία Diligent [21] έχει προταθεί στα πλαίσια υλοποίησης του ευρωπαϊκού ερευνητικού έργου SWAP. Η Diligent έχει ως στόχο να υποστηρίξει τους εμπειρογνώμονες ενός γνωστικού πεδίου (domain experts) μέσα από ένα κατανεμημένο περιβάλλον, να σχεδιάσουν, να αναπτύξουν και να εξελίξουν οντολογίες, με τη βοήθεια μιας προσέγγισης βασισμένης στη Θεωρία Ρητορικής Δομής (Rhetorical Structure Theory, RST). Αυτή η μεθοδολογία υποθέτει ότι η οντολογία που θα αναπτυχθεί δεν θα καλύπτει εξολοκλήρου όλο το εύρος γνώσης του πεδίου στο οποίο αναφέρεται. Αντιθέτως, υποθέτει ότι η οντολογία θα εξελιχθεί με το χρόνο και θα εφαρμοσθεί στις ανάγκες του κάθε χρήστη. Η μεθοδολογία Diligent αποτελείται από πέντε βασικές ενέργειες:

- Χτίσιμο (build). Χτίσιμο της αρχικής «κοινής» οντολογίας συνεργατικά (σε κατανεμημένο περιβάλλον) από τους μηχανικούς οντολογιών σε συνεργασία με τους εμπειρογνώμονες του γνωστικού πεδίου και τους χρήστες της οντολογίας.
- **Τοπική διαμόρφωση** (local adaptation). Μετά την ανάπτυξη της αρχικής οντολογίας, οι χρήστες είναι ελεύθεροι να την προσαρμόσουν ο καθένας στις δικές του ανάγκες. Η προσαρμογές και οι αλλαγές που ενδεχομένως να κάνει ένας χρήστης γίνονται τοπικά, στο μέρος της οντολογίας που χρησιμοποιεί ο ίδιος και δεν του επιτρέπεται να παρέμβει στην οντολογία που χρησιμοποιούν όλοι οι χρήστες. Η ομάδα που ελέγχει κεντρικά την οντολογία συλλέγει αιτήματα για αλλαγές στην κοινή οντολογία.

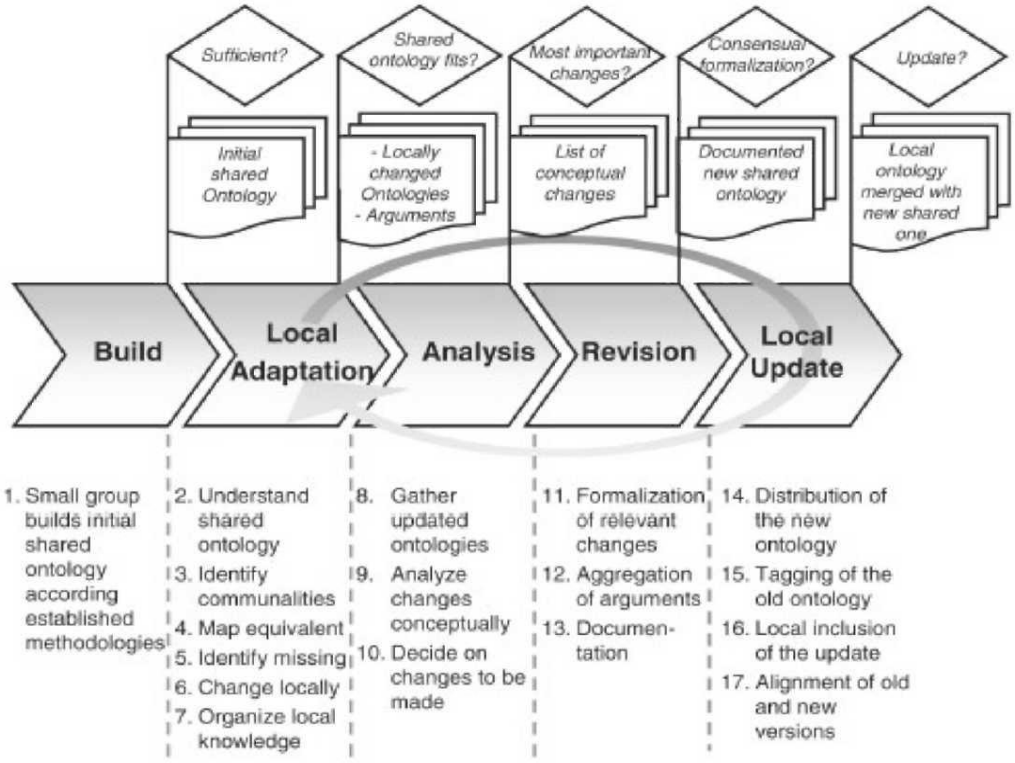
- **Ανάλυση** (analysis). Κατά τη φάση της ανάλυσης, η ομάδα ελέγχου της οντολογίας, αναλύει τις παρεμβάσεις που έχει κάνει ο κάθε χρήστης ξεχωριστά, καθώς και τα αιτήματα για αλλαγές στην κοινή οντολογία. Έπειτα καταλήγουν σε μία απόφαση για το ποιες από αυτές τις μετατροπές θα εφαρμοστούν στην κοινή οντολογία για να προκύψει η νέα έκδοσή της.

- Επανεξέταση (revision). Επανεξέταση και ενημέρωση της «κοινής» οντολογίας με τις συλλεγμένες αλλαγές και απαιτήσεις των χρηστών, που στοχεύει στην δημιουργία νέων εκδόσεων «κοινής» οντολογίας από τους μηχανικούς. Η ομάδα ελέγχου πρέπει να αναθεωρεί συχνά τις αλλαγές που γίνονται, έτσι ώστε οι τοπικές οντολογίες να μην απομακρυνθούν κατά πολύ από την κοινή οντολογία.

- Τοπική ενημέρωση (local update). Τοπική ενημέρωση των τοπικών οντολογιών των χρηστών με βάση τη νέα ενημερωμένη έκδοση «κοινής» οντολογίας.

Η διαδικασία ξεκινά με το χτίσιμο μιας αρχικής οντολογίας, της «κοινής» οντολογίας, από τους εμπειρογνώμονες του γνωστικού πεδίου, τους χρήστες, τους μηχανικούς γνώσης, και τους μηχανικούς οντολογιών. Στόχος είναι η ανάμειξη διαφορετικών ανθρώπων για κατανεμημένη ανάπτυξη οντολογιών, δηλαδή συμμετοχή διαφορετικών ομάδων χρηστών που έχουν διαφορετικές μεταξύ τους ανάγκες και σκοπούς, και που συνήθως δεν βρίσκονται στο ίδιο μέρος.

Στην εικόνα 3.3 παρουσιάζουμε γραφικά τις ενέργειες που περιλαμβάνει η μεθοδολογία Diligent που αναφέραμε παραπάνω.



Εικόνα 3.3 - Κύριες ενέργειες της μεθοδολογίας Diligent

4 Προτεινομενη Μεθοδολογια Περιγραφης Ασαφειας

Στο προηγούμενο κεφάλαιο παρουσιάσαμε μερικές από τις κυριότερες μεθοδολογίες που έχουν προταθεί και υιοθετηθεί από την επιστημονική κοινότητα για την περιγραφή και την ανάπτυξη οντολογιών. Στις μεθοδολογίες αυτές δεν γίνεται καμία αναφορά για το πώς μπορεί να αντιμετωπιστεί το πρόβλημα της ασάφειας που μπορεί να παρουσιάζει κάποιος ορός της οντολογίας. Όπως έχουμε ήδη αναφέρει ο όρος «Ασαφής Οντολογία» δημιουργήθηκε από την ανάγκη για την εξάλειψη του προβλήματος της ασάφειας που παρουσιάζεται κατά την αναπαράσταση γνώσης στον σημασιολογικό ιστό και συγκεκριμένα στις οντολογίες, αντιμετωπίζοντας το πρόβλημα αυτό με την χρήση της θεωρίας των ασαφών συνόλων.

Αυτό στο οποίο θα πρέπει να επικεντρωθεί περισσότερο η επιστημονική κοινότητα που ασχολείται με τις οντολογίες είναι η ανάπτυξη μεθοδολογιών περιγραφής ασαφών οντολογιών, οι οποίες θα εξασφαλίζουν την βελτίωση της διαδικασίας ανάπτυξης ασαφών οντολογιών αλλά και θα πιστοποιούσουν την ποιότητα του τελικού προϊόντος. Μία μεθοδολογία περιγραφής ασαφών οντολογιών θα πρέπει να εξασφαλίζει τα εξής:

- Αρχικά, θα πρέπει οι μηχανικοί οντολογιών και οι εμπειρογνώμονες της γνωστικής περιοχής στην οποία αναφέρεται η οντολογία, να είναι σε θέση να αναγνωρίζουν με εύκολο και ορθό τρόπο τα στοιχεία της οντολογίας τα οποία παρουσιάζουν ασάφεια.
- Οι μηχανικοί οντολογιών θα πρέπει να αποφασίζουν ποια είναι τα κατάλληλα ασαφή στοιχειά (Fuzzy Elements) για την περιγραφή της γνώσης.
- Οι εμπειρογνώμονες της γνωστικής περιοχής που αναφέρεται η οντολογία, θα πρέπει να μπορούν να αποφασίζουν κατά προσέγγιση αρχικά για τον βαθμό συμμέτοχης ενός στοιχείου σε ένα ασαφές σύνολο και στην συνεχεία να ορίζουν το βαθμό συμμετοχής ενός στοιχείου ανάλογα με το σενάριο χρήσης της οντολογίας, με την μεγαλύτερη δυνατή ακρίβεια.
- Τέλος, θα πρέπει η οντολογία που παράγεται να είναι επαναχρησιμοποιήσιμη, ο διαμοιρασμός να είναι εφικτός και στα στοιχεία στα οποία πραγματοποιήθηκε ασαφοποίηση να είναι καλά ορισμένα και κοινώς αποδεκτά.

Παραπάνω κάναμε λόγο για την ύπαρξη ασαφών στοιχείων (Fuzzy Elements) σε μία ασαφή οντολογία, χωρίς όμως να περιγράψουμε ποια είναι τα στοιχεία αυτά. Έτσι, στην συνεχεία για την καλύτερη κατανόηση του αναγνώστη, θα πραγματοποιηθεί μια συνοπτική περιγραφή των κύριων ασαφών στοιχείων μιας ασαφής οντολογίας καθώς και μία περιγραφή για τα είδη ασάφειας που μπορεί να παρουσιάζει μια ασαφής οντολογία. Τέλος θα γίνει μια λεπτομερής αναφορά για την μεθοδολογία περιγραφής ασάφειας σε μία οντολογία την «IKARUS - Onto»[22], στην οποία βασιστήκαμε για την δημιουργία της διαδικτυακής πλατφόρμας «*wb*GraphFuzzyOnto».

4.1 Είδη Ασαφειας

Η επιστημονική κοινότητα έχει ορίσει δυο βασικά είδη ασάφειας[22]:

- **Degree-Vagueness**. Μία ιδιότητα παρουσιάζει degree-vagueness λόγω της μη ύπαρξης ακριβών ορίων σχετικά με ένα μέγεθος. Για παράδειγμα για την ιδιότητα ψηλός για μία ομάδα καλαθοσφαίρισης δεν υπάρχουν ακριβή όρια για το μέγεθος ύψος. Δηλαδή μεταξύ ποιον διαστάσεων ύψους μπορεί να χαρακτηριστεί ένα παίχτης καλαθοσφαίρισης ψηλός. Θα πρέπει να επισημανθεί ότι μία ιδιότητα μπορεί να παρουσιάζει degree-vagueness για περισσότερα από ένα μεγέθη.

- **Combinatory-Vagueness.** Μία ιδιότητα παρουσιάζει combinatory-vagueness από την ύπαρξη πολλών ιδιοτήτων που χαρακτηρίζουν μια έννοια στην οποία αναφέρονται χωρίς όμως να υπάρχει μία σαφής διάκριση μεταξύ των ιδιοτήτων αυτών που να χαρακτηρίζουν την έννοια στην οποία αναφέρονται μεμονωμένα. Για παράδειγμα για την έννοια θρησκεία υπάρχουν αρκετές ιδιότητες την οποία την χαρακτηρίζουν (π.χ. πίστη σε υπερφυσικά όντα, τελετουργικές πράξεις), δεν υπάρχει όμως μια σαφής διάκριση η οποία να μας δείχνει σε τι βαθμό μία ιδιότητα είναι σε θέση να χαρακτηριστεί ως θρησκεία.

Στο σημείο είναι πολύ σημαντικό να αναφέρουμε ότι δεν θα πρέπει να συγχέεται ο όρος της ασάφειας με τους όρους της αβεβαιότητας και της ανακρίβειας καθώς επίσης και οι τιμές των βαθμών συμμετοχής ενός στοιχείου όπως έχουμε ήδη τονίσει και στο κεφάλαιο 2.5.1 ποικίλουν ανάλογα με το σενάριο χρήσης και με το περιβάλλον εφαρμογής.

4.2 Ασαφη Στοιχεια

Τα βασικά στοιχεία μιας ασαφής οντολογίας είναι παρόμοια με τα στοιχεία μιας συμβατικής οντολογίας (π.χ. Κλάσεις, Στιγμιότυπα, Σχέσεις, Ιδιότητες) με την μονή διάφορα ότι επιτρέπουν την έκφραση βαθμών συμμετοχής για την ασαφοποίηση των όρων της πληροφορίας που περιεχούν ασάφεια. Παρακάτω παρουσιάζουμε τα βασικά στοιχεία που περιλαμβάνει μιας ασαφής οντολογίας, όπως αυτά προταθήκαν από την μεθοδολογία IKARUS - Onto:

- **Ασαφείς Έννοιες (Fuzzy Concepts)**. Μία έννοια θεωρείται ασαφής αν τα στιγμιότυπα της συγκεκριμένης έννοιας ανήκουν στην έννοια αυτήν ως προς κάποιο βαθμό. Για παράδειγμα, αν υποθέσουμε ότι έχουμε την έννοια «ψηλός άνθρωπος» που η συγκεκριμένη έννοια περιέχει ασάφεια (ο ορός ψηλός είναι ασαφής) τότε θα μπορούσαμε να είχαμε την εξής πρόταση: Ο άνθρωπος y αποτελεί στιγμιότυπο της έννοιας (ή κλάσης) ψηλός άνθρωπος σε βαθμό 0,7.

- **Ασαφείς Σχέσεις (Fuzzy Relations).** Μία ασαφής σχέση σε μία πρόταση αντιστοιχεί στην ιδιότητα της πρότασης οπού συνδέει δυο στιγμιότυπα κάποιων κλάσεων και δηλώνει σε πιο βαθμό η πρόταση αυτή είναι αληθής. Έτσι, για

παράδειγμα η σχέση «είναι ειδικός» είναι ασαφής σχέση διότι περιέχει τον όρο ειδικός και θα μπορούσαμε να ορίσουμε την εξής πρόταση: «Ο άνθρωπος ψ είναι ειδικός στον τομέα ψηφιακά συστήματα σε βαθμό 0,7 ». Με την πρόταση αυτή δηλώνουμε σε πιο βαθμό ο άνθρωπος ψ είναι ειδικός στον τομέα ψηφιακά συστήματα.

- **Ασαφείς Ιδιότητες (Fuzzy Attributes)**. Μία ασαφής ιδιότητα διακατέχει τον ίδιο ρόλο με μία ασαφής σχέση σε μία πρόταση με την μόνη διαφορά ότι συνδέει ένα στιγμιότυπο μια κλάσης με ένα λεκτικό.

- **Ασαφείς Τύποι Δεδομένων (Fuzzy Datatypes).** Ένας ασαφής τύπος δεδομένων αντιστοιχεί σε ένα σύνολο ασαφών λεκτικών όρων που αποτελούν τις τιμές των ιδιοτήτων σε μια οντολογία. Ουσιαστικά πρόκειται για μία λεκτική αναπαράσταση των τιμών μιας ιδιότητας όπως είχαμε αναφέρει και στο κεφάλαιο 2.5.1 οπού μια τιμή μιας ιδιότητας ανήκει σε μια περιοχή τιμών που αντιστοιχεί σε κάποιο λεκτικό όρο ως προς κάποιο βαθμό.

4.3 Μεθοδολογια IKARUS-Onto

Η μεθοδολογία IKARUS-Onto[22] έχει σαν κύριο στόχο την βελτίωση της διαδικασίας ανάπτυξης ασαφών οντολογιών εξασφαλίζοντας όλα εκείνα τα χαρακτηριστικά που αναφέραμε παραπάνω και θα πρέπει να διέπουν μια μεθοδολογίας ανάπτυξης ασαφών οντολογιών.

Η IKARUS-Onto ορίζει ένα σύνολο από συγκεκριμένα βήματα και κατευθυντήριες γραμμές που θα πρέπει να ακολουθηθούν για την απόδοση της ασάφειας με την μορφή ασαφών χαρακτηριστικών. Η συγκεκριμένη μεθοδολογία δεν εστιάζει στην δομή μιας ασαφής οντολογίας αλλά στην διαδικασία που θα πρέπει να ακολουθηθεί για την επίτευξη της ασαφοποίησης των στοιχείων της οντολογίας οπού απαιτείται, με την μέγιστη δυνατή ακρίβεια. Θα πρέπει να επισημανθεί ότι IKARUS-Onto είναι μία μεθοδολογία κυρίως για την μετατροπή μιας συμβατικής (crisp) οντολογίας σε μια ασαφή οντολογία για αυτό και προαπαιτεί την ύπαρξη μιας συμβατικής οντολογίας. Για την υλοποίηση της συμβατικής οντολογίας μπορεί να χρησιμοποιηθεί οποιαδήποτε μεθοδολογία περιγραφής συμβατικών οντολογιών, που έχει προταθεί από την επιστημονική κοινότητα που ασχολείται με τις οντολογίες.

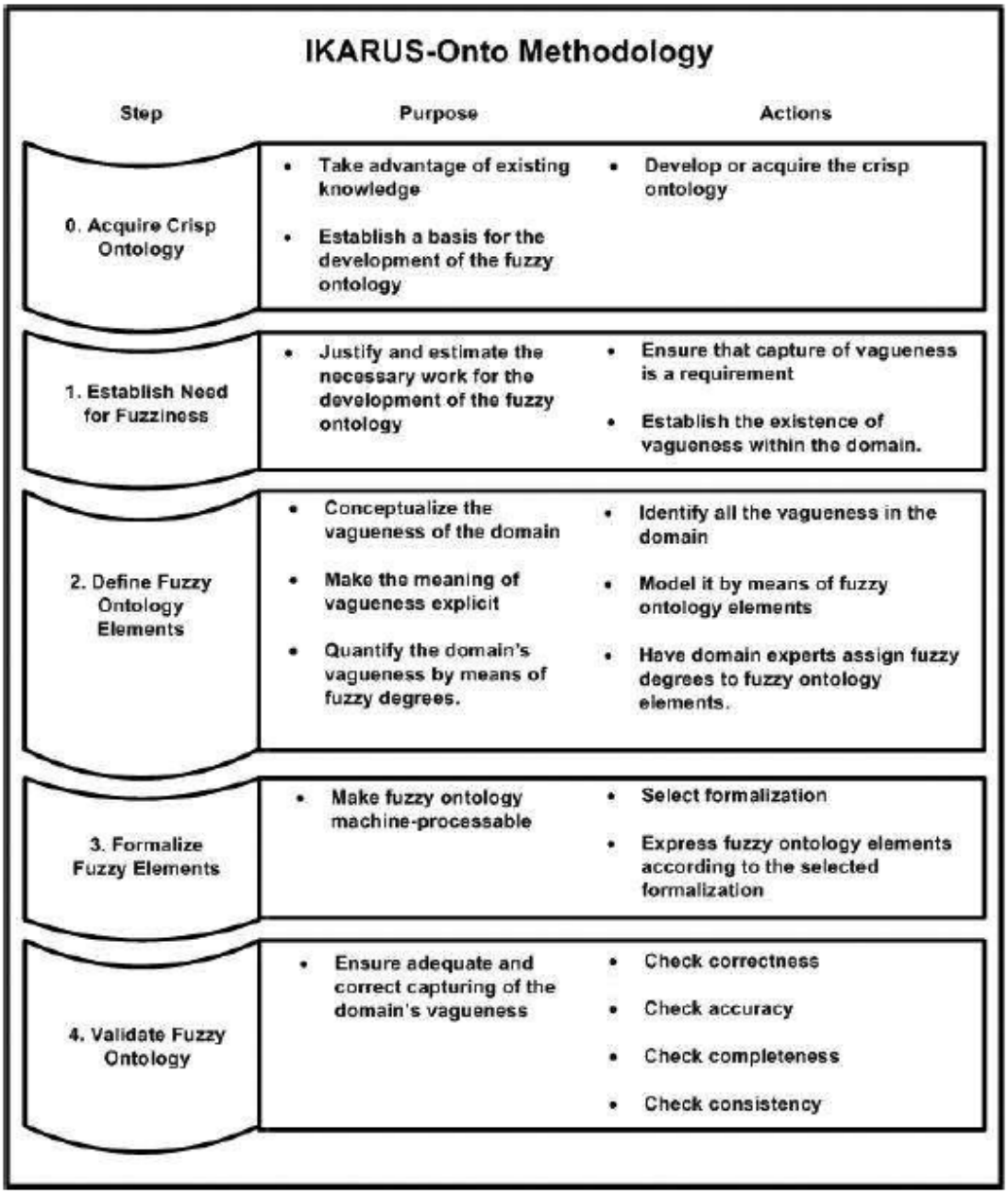
Συνιστάται αυτή η προσέγγιση δηλαδή της προϋπόθεσης ύπαρξης μιας συμβατικής οντολογίας διότι:

- Είναι πιο εύκολο για τους μηχανικούς οντολογιών και για τους εμπειρογνώμονες της συγκεκριμένης γνωστικής περιοχής να αναγνωρίσουν την πληροφορία που παρουσιάζει ασάφεια όταν αυτή έχει μοντελοποιηθεί και δομηθεί με έναν συγκεκριμένο τρόπο.

- Σε πολλές συμβατικές οντολογίες το τμήμα της οντολογίας που παρουσιάζει ασάφεια αποτελεί ξεχωριστό μέρος της οντολογίας. Αυτή η προσέγγιση δίνει

την δυνατότητα στους μηχανικούς οντολογιών να αναθέσουν ο ασαφές τμήμα της διαδικασίας ανάπτυξης ασαφών οντολογιών στους ειδικούς (εμπειρογνώμονες) της συγκεκριμένης γνωστικής περιοχής, οι οποίοι δεν είναι εξοικειωμένοι με τις συμβατικές μεθοδολογίες περιγραφής οντολογιών.

Στην εικόνα 4.1 παρουσιάζουμε τα κύρια σταδία της μεθοδολογίας IKARUS-Onto και τα οποία αναλύουμε στην συνέχεια.



Εικόνα 4.1- Σταδία Ανάπτυξης Μεθοδολογίας IKARUS-Onto

4.3.1 *Αποδειξη Υπαρξης Αναγκαιοτητας Ασαφοποιησης Στοιχειων*

Στο συγκεκριμένο στάδιο οι μηχανικοί οντολογιών θα πρέπει να βεβαιωθούν αρχικά για την αναγκαιότητα ασαφοποίησης της γνώσης που περιγράφεται στην συμβατική οντολογία, λαμβάνοντας υπόψη το περιβάλλον εφαρμογής αλλά και τα προτεινόμενα σενάρια χρήσης.

Μόλις οι μηχανικοί βεβαιωθούν για την ανάγκη ασαφοποίησης κάποιων στοιχείων της συγκεκριμένης γνωστικής περιοχής, θα πρέπει στην συνέχεια να προβούν στην αναγνώριση των στοιχείων αυτών. Συγκεκριμένα η διαδικασία που ακλουθείτε σε αυτό το στάδιο είναι η εξής:

- **Αναγνώριση Ασαφών Εννοιών.** Μία έννοια μπορεί να θεωρηθεί ασαφής, αν υπάρχουν στιγμιότυπα της συγκεκριμένης έννοιας δεδομένου της γνωστικής περιοχής και του σεναρίου χρήσης της οντολογίας και δεν είναι ξεκάθαρο αν τα στιγμιότυπα αυτά ανήκουν εξολοκλήρου στην έννοια αυτή ή όχι. Αρχικά υποψήφιες ασαφείς έννοιες μπορούν να χαρακτηριστούν οι έννοιες που δηλώνουν κάποια κατάσταση καθώς και οι έννοιες, όπου τα στιγμιότυπα τους αντικατοπτρίζουν ποιοτικές καταστάσεις (π.χ. ψηλός, κοντός).

- **Αναγνώριση Ασαφών Σχέσεων και Ιδιοτήτων.** Μία σχέση(η ιδιότητα που συνδέει δυο στιγμιότυπα σε μία σημασιολογική πρόταση) ή μία ιδιότητα (η ιδιότητα που συνδέει ένα στιγμιότυπο και έναν λεκτικό ορό σε μία σημασιολογική πρόταση) μπορούν να χαρακτηριστούν ασαφείς, αν δεδομένου του σεναρίου χρήσης της οντολογίας αλλά και της συγκεκριμένης γνωστικής περιοχής στην οποία αναφέρεται η οντολογία, υπάρχουν αμφιβολίες το κατά ποσό αληθής είναι η σημασιολογική πρόταση.

- **Αναγνώριση των Ασαφών Όρων μιας Ιδιότητας.** Η συγκεκριμένη διαδικασία περιλαμβάνει την αναγνώριση των λεκτικών όρων που αποτελούν τιμές σε μία ασαφή ιδιότητα.

Στο συγκεκριμένο στάδιο δεν απαιτείται μία λεπτομερής ανάλυση των ασαφών στοιχείων της συμβατικής οντολογίας ως προς την ασάφεια που παρουσιάζουν, απλώς απαιτείται η αναγνώριση όλων των στοιχείων που συντελούν στην ύπαρξη ασάφειας στην συμβατική οντολογία.

4.3.2 *Ορισμος – Περιγραφη Ασαφων Στοιχειων της Οντολογιας*

Το συγκεκριμένο στάδιο της μεθοδολογίας περιλαμβάνει μία λεπτομερής περιγραφή της ασαφούς γνώσης για τα στοιχεία που αναγνωριστήκαν στο προηγούμενο στάδιο και την απόδοση τους με όρους ασαφών στοιχείων όπως περιγράψαμε στην αρχή του κεφαλαίου. Σε αυτό το στάδιο πραγματοποιείται ουσιαστικά η ασαφοποίηση της οντολογίας. Κύριος στόχος αυτού σταδίου είναι ότι τα ασαφή στοιχεία που θα προκύψουν να είναι καλά ορισμένα και να επιτρέπουν την επαναχρησιμοποίηση τους καθώς και των

διαμοιρασμό τους. Σημαντικό είναι επίσης οι ασαφείς βαθμοί (αντίστοιχοι των βαθμών συμμετοχής που αναφέραμε στο κεφάλαιο 2.5.1 στα ασαφή σύνολα) για τα ασαφή στοιχεία στα οποία αναφέρονται να έχουν οριστεί κατά προσέγγιση με την μέγιστη δυνατή ακρίβεια.

Για τη επίτευξη όλων των παραπάνω στόχων που αναφέραμε, η IKARUS-Onto προτείνει μια συγκεκριμένη διαδικασία. Αρχικά προτείνει η διαδικασία να ξεκινήσει με την περιγραφή των ιδιοτήτων και σχέσεων της συμβατικής οντολογία που παρουσιάζουν ασάφεια και επιλέξαμε στο προηγούμενο βήμα. Αυτή διαδικασία περιλαμβάνει τις εξής ενέργειες με την εξής σειρά:

- Ο προσδιορισμός για κάθε σχέση και ιδιότητα του είδους ασάφειας (degree-vagueness, combinatory-vagueness) που παρουσιάζει κάθε στοιχείο και εξαιτίας ποιου μεγέθους.
- Στην συνέχεια θα πρέπει να πραγματοποιηθεί μια πιο λεπτομερής περιγραφή για το είδος της ασάφειας κάθε στοιχείου που ορίσαμε παραπάνω, συνάρτηση των μεγεθών που οφείλεται η ασάφεια. Αν ένα στοιχείο παρουσιάζει degree-vagueness εξαιτίας πολλών μεγεθών θα πρέπει να περιγραφεί η ασάφεια που παρουσιάζει ένα στοιχείο για το κάθε μέγεθος στο οποίο οφείλεται η ασάφεια.
- Ακλουθεί, ο προσδιορισμός της ερμηνείας σχετικά με τον βαθμό ασάφειας που συνδέει δυο στιγμιότυπα ή ένα στιγμιότυπο και έναν γλωσσικό όρο.
- Τέλος, ορισμός των βαθμών ασάφειας για τα συσχετιζόμενα στιγμιότυπα μιας σχέσης ή ο ορισμός των βαθμών ασάφειας ενός στιγμιότυπου και ενός λεκτικού μιας ιδιότητας. Οι βαθμοί θα πρέπει να οριστούν κατά προσέγγιση με την μέγιστη δυνατή ακρίβεια.

Η περιγραφή του είδους της ασάφειας κάθε στοιχείου(στην συγκεκριμένη περίπτωση σχέσης ή ιδιότητας) αλλά και της ερμηνεία των βαθμών ασάφειας μεταξύ των εμπλεκόμενων μερών συμβάλουν στην καλύτερη κατανόηση της ασαφούς γνώσης και παρέχεται έτσι η δυνατότητα στους εμπειρογνώμονες να ορίσουν τους βαθμούς ασάφειας με μεγαλύτερη ακρίβεια. Στην συνέχεια παραθέτουμε ένα παράδειγμα για την ασαφή ιδιότητα «είναι ειδικός». Έστω ότι έχουμε την ακόλουθη πρόταση «Ο άνθρωπος ψ είναι ειδικός στον τομέα ψηφιακά συστήματα» η οποία παρουσιάζει ασάφεια.

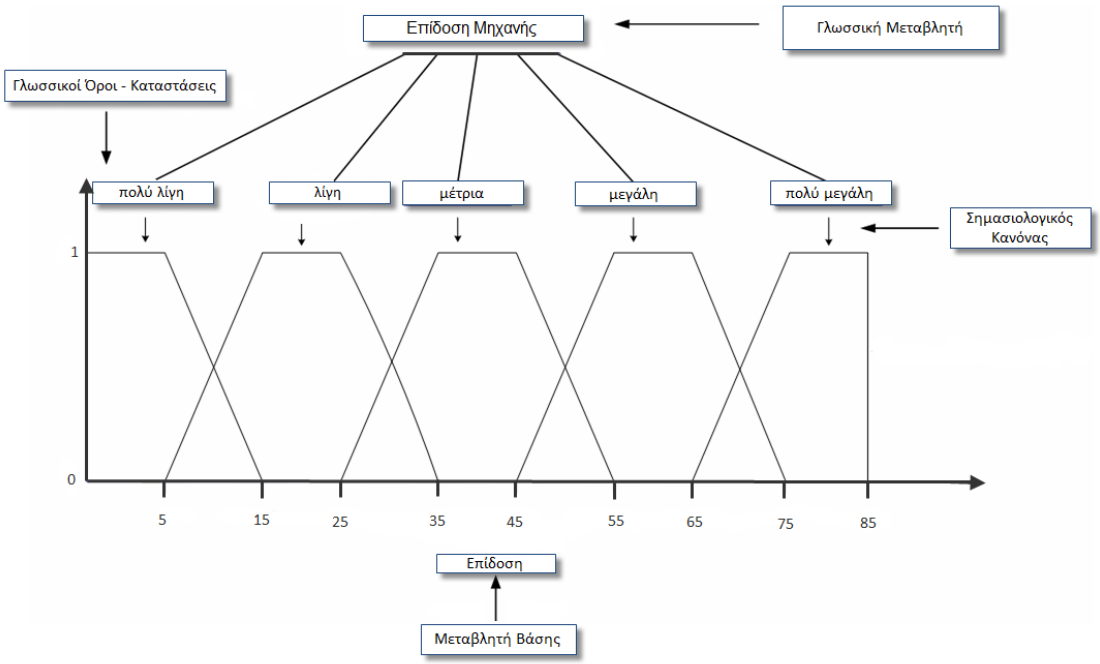
Attribute	**Vagueness Nature**	**Degree Interpretation**
είναι ειδικός	**Degree-vagueness εξαιτίας του μεγέθους ειδικός**	**Σε τι βαθμό είναι ειδικός ο άνθρωπος ψ με τον τομέα ψηφιακά συστήματα.**

Πίνακας 4.1 – Περιγραφή ασαφής ιδιότητας

Μετά την ολοκλήρωση της διαδικασίας περιγραφής των ασαφών σχέσεων και ιδιοτήτων ακολουθεί η περιγραφή των ασαφών τύπων δεδομένων (fuzzy datatypes). Η συγκεκριμένη διαδικασία περιλαμβάνει τις εξής ενέργειες:

- Την αναγνώριση όλων των ασαφών ιδιοτήτων οπού οι τιμές τους αποδίδονται μέσω ασαφών λεκτικών ορών.
- Μετά την αναγνώριση των ορών αυτών πραγματοποιείται η ομαδοποίηση τους αν ιδιότητα στην οποία αναφέρονται.
- Ορίζουμε για το κάθε γλωσσικό όρο για την ιδιότητα στην οποία αναφέρονται ένα σύνολο τιμών από το οποίο θα προκύψει και ο βαθμός συμμετοχής μιας τιμής χ σε ένα ασαφές σύνολο.

Είναι πολύ σημαντικό να αναφέρουμε ότι ένας ασαφές όρος μπορεί να αποτελεί τιμή σε περισσότερες από μία ιδιότητες. Η παραπάνω διαδικασία είναι εφάμιλλη της διαδικασίας ασαφοποίησης για την γλωσσική μεταβλητή (στην περίπτωση μας ιδιότητα) «Απόδοση Μηχανής» που περιγράψαμε στο κεφάλαιο 2.5.1.



Εικόνα 4.2 - Διάγραμμα γλωσσικής μεταβλητής «Απόδοσης Μηχανής»

Αυτό το στάδιο της μεθοδολογίας ολοκληρώνεται με τον ορισμό των ασαφών εννοιών (fuzzy concepts). Η διαδικασία που ακολουθείται είναι εφάμιλλη της διαδικασίας περιγραφής ασαφών σχέσεων και ιδιοτήτων με μία βασική διάφορα όμως. Πολλές φορές ένα στιγμιότυπό μίας έννοιας οφείλει την ασάφεια του εξαιτίας κάποιας ασαφής σχέσης ή ιδιότητας ή εξαιτίας κάποιου ασαφή λεκτικού ορού, σε αυτές τις περιπτώσεις ο ορισμός και η περιγραφή των ασαφών εννοιών προκύπτει από το στοιχείο που περιέχει την ασάφεια. Η διαδικασία περιγραφής των ασαφών εννοιών περιλαμβάνει τις εξής ενέργειες:

- Την αναγνώρισή όλων των εννοιών της συμβατικής οντολογίας οι οποίες παρουσιάζουν ασάφεια, συμφωνά με τις κατευθυντήριες γραμμές του πρώτου σταδίου.

- Την εξακρίβωση αν η ασάφεια για ένα στιγμιότυπο μιας έννοιας οφείλεται σε κάποιο άλλο στοιχείο όπως αναφέραμε παραπάνω.
- Σε περίπτωση που η ασάφεια δεν προκύπτει από κάποιο άλλο στοιχείο , θα πρέπει να ακολουθηθούν τα ιδία βήματα που αναφέραμε στις περιπτώσεις των ασαφών σχέσεων και ιδιοτήτων.

4.3.3 *ΔΙΑΤΥΠΩΣΗ ΤΩΝ ΑΣΑΦΩΝ ΣΤΟΙΧΕΙΩΝ ΜΕ ΜΙΑ ΣΗΜΑΣΙΟΛΟΓΙΚΗ ΑΣΑΦΗ ΓΛΩΣΣΑ*

Σε αυτό το στάδιο της διαδικασία ανάπτυξης ασαφών οντολογιών, πραγματοποιείται η περιγραφή με μία σημασιολογική γλώσσα περιγραφής ασάφειας, των ασαφών στοιχείων που πρόεκυψαν στο προηγούμενο βήμα. Η επιλογή της γλώσσα που θα χρησιμοποιήσουμε για να περιγράψουμε τα ασαφή στοιχεία διαφέρει ανάλογα με τις χαρακτηριστικά των ασαφών στοιχείων. Επισημαίνεται ότι θα πρέπει να έχει πραγματοποιηθεί έλεγχος σχετικά με τα χαρακτηριστικά της κάθε γλώσσας ώστε να μην παρουσιαστεί κάποια αδυναμία στην αναπαράσταση της ασαφούς γνώσης. Συγκεκριμένα θα πρέπει να δοθεί μεγάλη σημασία στα ασαφή χαρακτηριστικά (fuzzy elements) τα οποία υποστηρίζει μια σημασιολογική γλώσσα καθώς επίσης και στις ασαφείς πράξεις (π.χ. συνεπαγωγή) που θα πρέπει να υποστηρίζονται από μία σημασιολογική γλώσσα.

Εν κατακλείδι, το συμπέρασμά που προκύπτει από την παραπάνω παράγραφο, ότι ανάλογα με την γνωστική περιοχή στην οποία αναφέρεται μία οντολογία σε συνδυασμό πάντα με προτεινόμενο σενάριο χρήσης, μια σημασιολογική γλώσσα θα είναι καταλληλότερη για την αναπαράσταση ασαφούς γνώσης σε σχέση με κάποια άλλη.

4.3.4 *ΈΛΕΓΧΟΣ ΕΓΚΥΡΟΤΗΤΑΣ ΑΣΑΦΗΣ ΟΝΤΟΛΟΓΙΑΣ*

Μόλις ολοκληρωθεί η ανάπτυξή της οντολογίας, θα πρέπει να εκτελεστεί μία διαδικασία έλεγχου της εγκυρότητας της ασαφής οντολογίας η οποία να εξασφαλίζει ότι αναπαραστάθηκε η ασαφής γνώσης με ορθό τρόπο. Τα βασικά χαρακτηριστικά που θα πρέπει να διέπουν μία ασαφή οντολογία είναι τα ακόλουθα:

- **Εγκυρότητα.** Μια οντολογία χαρακτηρίζεται έγκυρη αν έχει πραγματοποιηθεί ασαφοποίηση μόνο στα στοιχεία που παρουσιάζουν ασάφεια και όχι σε αλλά στοιχεία.
- **Ακρίβεια.** Μία ασαφής οντολογία θεωρείται ακριβής αν οι βαθμοί ασάφειας των ασαφών στοιχείων μιας οντολογίας, έχουν αποδοθεί με την μέγιστη δυνατή ακρίβεια, λαμβάνοντας υπόψη την γνωστική περιοχή στην οποία αναφέρονται και τα προτεινόμενα σενάρια χρήσης. Αυτό δεν σημαίνει ότι στους βαθμούς ασάφειας θα πρέπει να έχουν εκχωρηθεί συγκεκριμένες τιμές, αλλά ότι δεν θα πρέπει να έρχονται σε αντίθεση με την γνωστική περιοχή στην οποία αναφέρονται. Για παράδειγμα η πρόταση «Ο *Μπιλ* Γκέιτς είναι πλούσιος σε βαθμό 0,1 » είναι ανακριβής.

- **Πληρότητα.** Μια ασαφής οντολογία χαρακτηρίζεται πλήρης, αν έχουν ασαφοποιηθεί όλοι οι όροι της συμβατικής οντολογίας που περιείχαν ασάφεια για την γνωστική περιοχή που αναφέρεται η οντολογία και βάση των προτεινόμενων σεναρίων χρήσης.

- **Συνέπεια.** Συνεπής χαρακτηρίζεται μία οντολογία που δεν περιέχει αντικρουόμενες ερμηνείες για τους βαθμούς ασάφειας των ασαφών στοιχείων που αναφέρονται στην ιδία ασαφή γνώση και εξάγουν προφανώς λανθασμένα συμπεράσματα. Οι προτάσεις «Ο άνθρωπος ψ είναι λευκός σε βαθμό 0,2» και «Ο άνθρωπος ψ έχει λευκό δέρμα σε βαθμό 0,9» είναι αντικρουόμενες.

5 ΑΝΑΛΥΣΗ ΑΠΑΙΤΗΣΕΩΝ

Το παρόν κεφάλαιο περιγράφει όλες τις απαραίτητες πληροφορίες σχετικά με τις δυνατότητες και τις υπηρεσίες που θα παρέχονται σε κάθε χρήστη. Σκοπός του λοιπόν, είναι ο προσδιορισμός των υπηρεσιών αυτών αλλά και των δυνατοτήτων κάθε χρήστη.

5.1 ΟΡΙΣΜΟΙ

- **Μηχανικοί Ασαφών Οντολογιών**. Με τον όρο «Μηχανικοί Ασαφών Οντολογιών» αναφερόμαστε στην ομάδα χρηστών οι οποίοι είναι προγραμματιστές – μηχανικοί λογισμικού και ειδικοί στην ανάπτυξη και περιγραφή οντολογιών. Επίσης εκτός από την περιγραφή οντολογιών είναι σε θέση να αναγνωρίζουν τα στοιχεία που αναπαριστούν ασαφή γνώση σε μία οντολογία.

- **Ειδικοί (Εμπειρογνώμονες) Γνωστικής Περιοχής.** Με τον όρο «Ειδικοί γνωστικής περιοχής» αναφερόμαστε στην ομάδα χρηστών η οποία κατέχει την απαιτούμενη γνώση που αφορά μια γνωστική περιοχή. Συγκεκριμένα για την ανάπτυξη μιας οντολογίας, δεδομένου ενός συγκεκριμένου σεναρίου χρήσης, των χρηστών που συμμετέχουν και λαμβάνοντας υπόψη όλες τις μεταβλητές που είναι άρρηκτα συνδεμένες με το συγκεκριμένο σενάριο χρήσης μια γνωστικής περιοχής, είναι αυτοί που μπορούν αποδώσουν κατά προσέγγιση και με την μεγίστη δυνατή ακρίβεια τους βαθμούς ασάφειας ενός στοιχείου.

- **Vagueness Nature.** Με τον συγκεκριμένο όρο αναφερόμαστε στην περιγραφή του είδους της ασάφειας (Degree-Vagueness ή Combinatory- Vagueness) που μπορεί να υπάρχει στην απόδοση μιας πληροφορίας εξαιτίας ενός συγκειμένου μεγέθους. Αποτελεί ένα είδος σχολίων για την καλύτερη κατανόηση της ασάφειας.

- **Degree Interpretation.** Με τον συγκεκριμένο όρο αναφερόμαστε σε μια πιο λεπτομερής περιγραφή –επεξήγηση των βαθμών ασάφειας και των εμπλεκόμενων μερών. Ουσιαστικά πρόκειται για μια καταγραφή των εμπλεκόμενων μερών που συμβάλουν στην ύπαρξη ασάφειας σε μία πρόταση, δηλαδή του μεγέθους που οφείλεται η ύπαρξη της ασάφειας, και των στιγμιότυπων μια κλάσης που συμμετέχουν στην πρόταση ή των λεκτικών που έχουν τον ρόλο ενός αντικείμενου σε μία πρόταση. Αποτελεί όπως και στην παραπάνω περίπτωση ένας είδος σχολίων για την καλύτερη κατανόηση της ασάφειας και με απώτερο στόχο την συμβολή σε μία πιο ακριβέστερη απόδοση των βαθμών ασάφειας.

5.2 ΒΑΣΙΚΕΣ ΛΕΙΤΟΥΡΓΙΕΣ ΣΥΣΤΗΜΑΤΟΣ

Στην παρούσα ενότητα θα προσδιορίσουμε τις βασικές λειτουργίες της πλατφόρμας «*wb*GraphFuzzyOnto» που αναπτύχθηκε, ανάλογα με την ομάδα χρηστών στην οποία απευθύνεται.

Πριν πραγματοποιηθεί αναλυτική περιγραφή των υπηρεσιών που παρέχονται σε κάθε ομάδα χρηστών και για την καλύτερη κατανόηση του αναγνώστη, κρίνεται απαραίτητο να δοθούν κάποιες επισημάνσεις όσο αφορά τον τρόπο λειτουργίας της πλατφόρμας «*wb*GraphFuzzyOnto».

Όπως έχουμε ήδη αναφέρει, η πλατφόρμα ασαφοποίησης που αναπτύχθηκε, βασίζεται στην μεθοδολογία «IKARUS-Onto» και συγκεκριμένα στα βήματα που περιγράφονται στα κεφάλαια 4.3.1 (Απόδειξη Ύπαρξης Αναγκαιότητας Ασαφοποίησης Στοιχείων) και 4.3.2 (Ορισμός – Περιγραφή Ασαφών Στοιχείων της Οντολογίας). Στόχος της λοιπόν είναι, η εύρεση όλων των κυρίων στοιχείων που απαρτίζουν μία οντολογία (στιγμιότυπα εννοιών, σχέσεις, ιδιότητες και λεκτικά) που αποδίδουν ασαφή γνώση και να αποδοθούν τα στοιχεία αυτά με όρους ασαφών στοιχείων(Fuzzy Elements).

Στην συνέχεια ακολουθούν οι βασικές λειτουργίες της πλατφόρμας ανά ομάδα χρηστών και με τα αντίστοιχα διαγράμματα περιπτώσεων χρήσης (Use Cases):

Ομάδα χρηστών: **Μηχανικοί Οντολογιών**

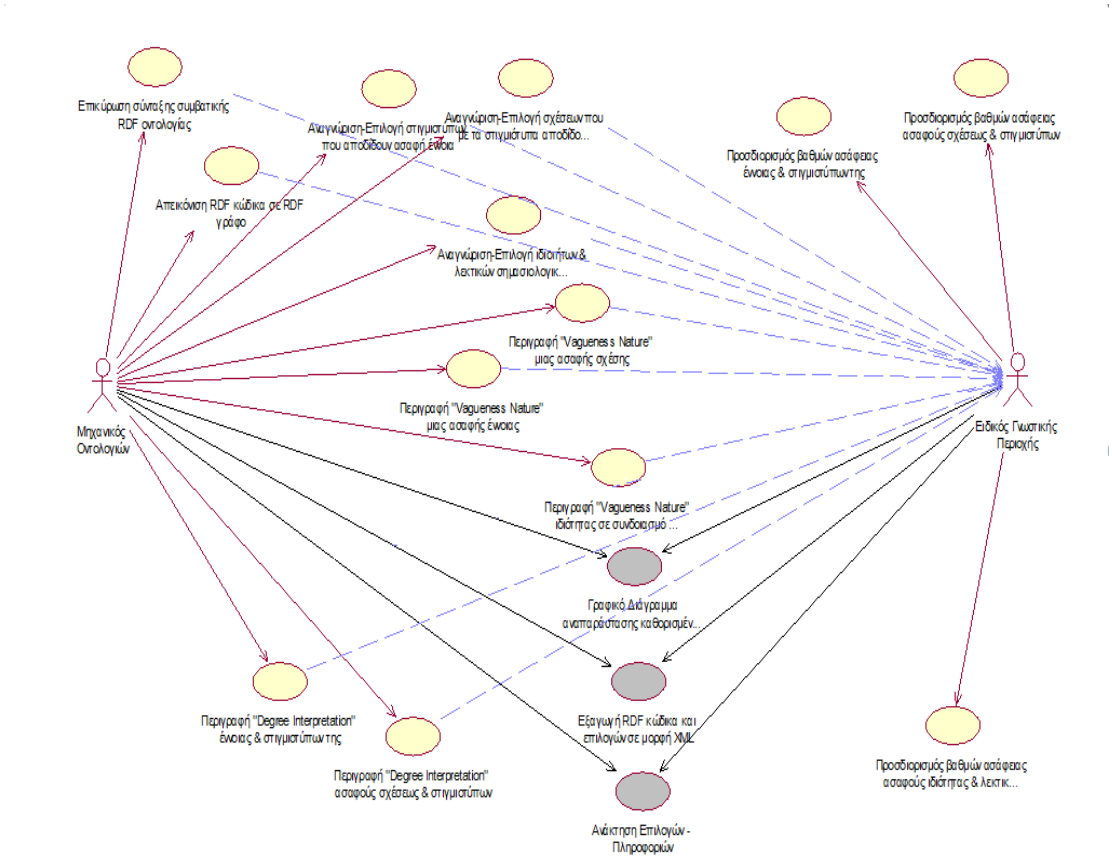
- Επικύρωση της ορθής και έγκυρης σύνταξης μιας συμβατικής οντολογίας υλοποιημένης σε RDF σημασιολογική γλώσσα.
- Απεικόνιση του RDF κώδικα σε μορφή RDF γράφου ενσωματώνοντας την κατάλληλη διαδραστικότητα.
- Δυνατότητα αναγνώρισης και επιλογής των στιγμιότυπων μιας έννοιας(κλάσης) που αποδίδουν ασαφή γνώση.
- Δυνατότητα αναγνώρισης και επιλογής των σχέσεων, όπου μαζί με τα εμπλεκόμενα στοιχεία(στιγμιότυπα εννοιών) σε μια σημασιολογική πρόταση αποδίδουν ασαφή γνώση.
- Δυνατότητα αναγνώρισης και επιλογής των ιδιοτήτων και λεκτικών μιας σημασιολογικής πρότασης. Ένα λεκτικό σε μία πρόταση όπως έχουμε ήδη αναφέρει αποτελεί τιμή της ιδιότητας και το συγκεκριμένο λεκτικό αναπαριστά ασαφή γνώση.
- Περιγραφή του «**Vagueness Nature**» για μια ασαφή σχέση.
- Περιγραφή του «**Vagueness Nature**» για μια ασαφή έννοια.
- Εμφάνιση γραφικού διαγράμματος αναπαράστασης των καθορισμένων ορίων(πεδίο τιμών) ενός λεκτικού που αναπαριστά ασαφή γνώση για μια ιδιότητα, αλλά και συνολική αναπαράσταση όλων των λεκτικών που αναπαριστούν ασαφή γνώση και αποτελούν τιμές μίας συγκεκριμένης ιδιότητας.
- Δυνατότητα περιγραφής του «**Degree Interpretation**» για μία ασαφή σχέση και των εμπλεκόμενων στιγμιότυπων μιας έννοιας ή και περισσότερων εννοιών.

- Δυνατότητα περιγραφής του «**Degree Interpretation**» για μία έννοια και των στιγμιότυπων αυτής της έννοιας.
- Εξαγωγή του RDF κώδικα και όλων των επίλογων και των πληροφοριών που έχουν πραγματοποιηθεί και οριστεί αντίστοιχα σε μορφή XML.
- Ανάκτηση όλων των επιλόγων και πληροφοριών που έχουν πραγματοποιηθεί και οριστεί.

Ομάδα χρηστών: **Ειδικοί Γνωστικής Περιοχής**

- Πρόσβαση σε όλες τις επιλογές (σχέσεων, στιγμιότυπων εννοιών και ιδιοτήτων) και πληροφορίες που έχουν πραγματοποιήσει και ορίσει αντίστοιχα οι μηχανικοί οντολογιών και περιγράψαμε παραπάνω.
- Προσδιορισμός των βαθμών ασάφειας για μια ασαφή σχέση και των εμπλεκόμενων στιγμιότυπων εννοιών.
- Προσδιορισμός των βαθμών ασάφειας για μια έννοια και των στιγμιότυπων της έννοιας αυτής.
- Προσδιορισμός των ορίων (πεδίο τιμών) ενός λεκτικού που αναπαριστά ασαφή γνώση ως τιμή μιας ιδιότητας, συμπληρώνοντας τα πεδία που απαιτούνται για την δημιουργία ενός γραφήματος όπως της Εικόνας 4.2 - Διάγραμμα γλωσσικής μεταβλητής «Απόδοσης Μηχανής».
- Εμφάνιση γραφικού διαγράμματος αναπαράστασης των καθορισμένων ορίων(πεδίο τιμών) ενός λεκτικού που αναπαριστά ασαφή γνώση για μια ιδιότητα, αλλά και συνολική αναπαράσταση όλων των λεκτικών που αναπαριστούν ασαφή γνώση και αποτελούν τιμές μίας συγκεκριμένης ιδιότητας.
- Εξαγωγή του RDF κώδικα και όλων των επίλογων και των πληροφοριών που έχουν πραγματοποιηθεί και οριστεί αντίστοιχα σε μορφή XML.
- Ανάκτηση όλων των επιλόγων και πληροφορίων που έχουν πραγματοποιηθεί και οριστεί.

Στην εικόνα 5.1 παρουσιάζονται οι περιπτώσεις χρήσεως για τις υπηρεσίες και τις ομάδες χρηστών που περιγράφτηκαν παραπάνω. Με την διακεκομμένες γραμμές επισημαίνεται η πρόσβαση της ομάδας χρηστών «Ειδικοί Γνωστικής Περιοχής» σε όλες τις πληροφορίες που έχουν ορίσει η ομάδα χρηστών «Μηχανικοί Οντολογιών». Επίσης με γκρι χρώμα σημειώνονται εκείνες οι περιπτώσεις χρήσεως (Use Cases) που είναι κοινές και για τις δύο ομάδες χρηστών.



Εικόνα 5.1 – Περιπτώσεις χρήσης

5.3 ΓΕΝΙΚΟΙ ΠΕΡΙΟΡΙΣΜΟΙ

5.3.1 *ΠΕΡΙΟΡΙΣΜΟΙ ΔΙΑΔΙΚΤΥΟΥ*

Μία από τις μεγαλύτερες προκλήσεις τις οποίες έχουν να αντιμετωπίσουν όλα τα διαδικτυακά συστήματα είναι να ικανοποιούν τις απαιτήσεις των χρηστών για γρήγορα αποτελέσματα στις αιτήσεις που κάνουν. Αν και πολλές φορές υπεύθυνο για τέτοια προβλήματα είναι το ίδιο το εύρος ζώνης (bandwidth) του δικτύου, ο υπεύθυνος ανάπτυξης συστημάτων θα πρέπει να προσπαθεί να κάνει όσο το δυνατό πιο αποδοτικό το σύστημά του. Τα χαρακτηριστικά που επηρεάζουν την απόδοση ενός online συστήματος είναι τα εξής:

- Μέγεθος δυναμικής και στατικής σελίδας
- Μέγεθος εικόνων που χρησιμοποιούνται στο layout
- Χρόνος απόκτησης των δεδομένων
- Χρόνος επεξεργασίας των δεδομένων

Στην πλατφόρμας «*wb*GraphFuzzyOnto» γίνεται χρήση πολύπλοκων αλγορίθμων, όπως ο αλγόριθμος δημιουργίας RDF γράφων. Απαίτηση του συστήματος είναι οι αλγόριθμοι αυτοί να είναι ικανοποιητικά γρήγοροι, έτσι ώστε ο χρήστης να λαμβάνει

γρήγορα τις συστάσεις του συστήματος. Καταβλήθηκε επίσης μεγάλη προσπάθεια στο να είναι η πλατφόρμα συναφές με τις αρχές που ορίζει ο τομέας της επικοινωνίας ανθρώπου μηχανής για την ανάπτυξη διεπαφών.

6 Επιλογη Τεχνολογιων

Η επιλογή των τεχνολογιών που θα χρησιμοποιηθούν για την υλοποίηση οποιουδήποτε έργου είναι μια απόφαση στρατηγικής σημασίας. Η απόφαση αυτή επηρεάζει την ποιότητα, την αξιοπιστία, την ευελιξία και την δυνατότητα συντήρησης του.

Στις παραγράφους που ακολουθούν πραγματοποιείται μια συνοπτική περιγραφή των τεχνολογιών που χρησιμοποιηθήκαν για την υλοποίηση της πλατφόρμας «*wb*GraphFuzzyOnto». Πραγματοποιείται μια συνοπτική περιγραφή των τεχνολογιών, διότι μία αναλυτική περιγραφή θα αποπροσανατολίσει τον αναγνώστη από τους στόχους και τον σκοπό της παρούσας διπλωματικής εργασίας.

Όπως έχουμε ήδη αναφέρει η υλοποίηση της πλατφόρμας «*wb*GraphFuzzyOnto» βασίστηκε στις τεχνολογίες Web2.0. Οι κύριοι λόγοι για το οποίο επιλέχτηκαν τεχνολογίες βασισμένες σε Web 2.0, είναι οι δυνατότητες που προσφέρονται για ανάπτυξη διεπαφών με πλούσιες δυνατότητες αλληλεπίδρασης με τον χρήστη, αλλά κυρίως και του γεγονότος ότι τα αποτελέσματα της ενέργειας του χρήστη ενημερώνουν μόνο τα κομμάτια της σελίδας που πρόκειται να τροποποιηθούν, χωρίς την επιβάρυνση του χρόνου που συνεπάγεται η επαναφόρτιση της σελίδας. Αυτό συμβάλει στο να είναι η λειτουργικότητα μιας διαδικτυακής εφαρμογής εφάμιλλη των δυνατοτήτων μιας εφαρμογής Desktop αλλά και σύγχρονος να παρέχονται όλες οι δυνατότητες μιας διαδικτυακής εφαρμογής (π.χ. απομακρυσμένη σύνδεση, μεγαλύτερη συμμετοχή χρηστών κ.α.). Επίσης η χρησιμοποίηση δικτυακών εφαρμογών με χαρακτηριστικά Desktop εφαρμογής, ενδείκνυται για άτομα τα όποια δεν είναι εξοικειωμένα με τον παγκόσμιο ιστό, που στην περίπτωση μας μπορεί να τυγχάνει να είναι οι ειδικοί (εμπειρογνώμονες) μίας γνωστικής περιοχής.

6.1 Πλατφορμα J2EE

Στις μέρες μας ολοένα και περισσότεροι προγραμματιστές ενδιαφέρονται να αναπτύξουν διαδικτυακές εφαρμογές, που χαρακτηρίζονται από συνεχώς αυξανόμενες δυνατότητες σε ταχύτητα, ασφάλεια και αξιοπιστία. Στον σημερινό γρήγορα εξελισσόμενο και απαιτητικό κόσμο της κοινωνίας της πληροφορίας, οι εφαρμογές αυτές πρέπει να σχεδιάζονται, να αναπτύσσονται και τελικά να παράγονται με λιγότερα χρήματα, με μεγαλύτερη ταχύτητα, και με λιγότερους πόρους φυσικά, απ’ ότι στο παρελθόν.

Τα παραπάνω κριτήρια ικανοποιούν σε μεγάλο βαθμό οι πλατφόρμες Java 2 Platform Standard Edition (J2SE) και Java 2 Platform Enterprise Edition (J2EE)[26]. Συγκεκριμένα, η J2EE πλατφόρμα προσφέρει ένα πολυεπίπεδο κατανεμημένο μοντέλο, επαναχρησιμοποιήσιμα τμήματα, ενοποιημένο μοντέλο ασφάλειας, ευέλικτο έλεγχο ασφάλειας και υποστήριξη υπηρεσιών ιστού δια μέσου ενός ολοκληρωμένου συστήματος ανταλλαγής δεδομένων βασισμένου στην γλώσσα XML (eXtensible Markup Language). Για τους παραπάνω λόγους αποφασίστηκε να χρησιμοποιήσουμε τεχνολογίες που βασίζονται κυρίως στην πλατφόρμα J2EE.

6.2 JAVA SERVLETS

Η τεχνολογία Java Servlets[31] αναφέρεται στον μηχανισμό που παρέχει η Java για επέκταση και ενίσχυση της λειτουργικότητας ενός διακομιστή και στην δυνατότητα πρόσβασης σε απομακρυσμένα ολοκληρωμένα συστήματα. Ουσιαστικά πρόκειται για μια κλάση της γλώσσας προγραμματισμού Java που χρησιμοποιείται για να επεκτείνει τις δυνατότητες πρόσβασης των εφαρμογών των εξυπηρετητών και των εξυπηρετούμενων μέσω ενός μοντέλου αιτήσεων – απαντήσεων. Αν και οι servlets κλάσεις μπορούν να ανταποκρίνονται σε κάθε είδος αιτήσεων, κυρίως χρησιμοποιούνται για να επεκτείνουν τις εφαρμογές που φιλοξενούνται από εξυπηρετητές του ιστού. Για τέτοιες εφαρμογές, η τεχνολογία Java servlet καθορίζει HTTP-εξειδικευμένες servlet κλάσεις.

Τα πακέτα javax.servlet και javax.servlet.http παρέχουν interfaces και κλάσεις με τις οποίες μπορούμε να γράψουμε servlets. Κάθε κλάση servlet θα πρέπει να υλοποιεί το Interface Servlet, που καθορίζει τις μεθόδους που ορίζουν με τη σειρά τους τον κύκλο ζωής ενός servlet.

Στην «*wb*GraphFuzzyOnto» πλατφόρμα χρησιμοποιήθηκαν servlet κλάσεις για την διαχείριση των αιτημάτων ενός χρήστη από τις επιλογές που πραγματοποιεί σε έναν rdf γράφο που απεικονίζεται μέσα από ένα SVG αρχείο. Τα δεδομένα που συλλέγονται από μία κλάση servlet, στην συνέχεια διαχειρίζονται αναλόγως μέσα από μία Java κλάση που διαχειρίζεται τα components μιας ZK (.zul) σελίδας.

6.3 JAVASCRIPT

Είναι μία γλώσσα scripting βασισμένη στη έννοια του πρότυπου προγραμματισμού και πήρε το όνομα της από την εφαρμογή ECMAScript του Netscape. Η γλώσσα αυτή χρησιμοποιείται στις ιστοσελίδες για διάφορους λόγους, όπως για δυναμική σχεδίαση, έλεγχο φόρμας και πολλά άλλα ουσιαστικά συμβάλει στην διαδραστικότητα με τον χρήστη.

Η JavaScript είναι μια «client-side» τεχνολογία δηλαδή εκτελείται από τον περιηγητή(browser) του πελάτη. Συγκεκριμένα, όταν ένας περιηγητής συναντήσει εντολές scripting ενσωματωμένες σε μια σελίδα HTML, εκχωρεί την ευθύνη της ερμηνείας του κώδικα στη μηχανή scripting που διαθέτει. Το τμήμα αυτό του πελάτη διερμηνεύει τις εντολές και είτε τις εκτελεί ή τις αποθηκεύει για μελλοντική χρήση.

Στην πλατφόρμα «*wb*GraphFuzzyOnto» χρησιμοποιούμε την JavaScript ενσωματωμένη σε SVG αρχεία για να διαχειριστούμε συγκεκριμένα γεγονότα(events) που μπορεί να προκαλέσει ο χρήστης, όπως κάποιες επιλογές που μπορεί να πραγματοποιήσει πάνω σε έναν γράφο. Στην συνεχεία με κατάλληλες Ajax μεθόδους αποστέλλουμε δεδομένα στην πλευρά του διαχειριστή όπου επεξεργάζονται από κλάσεις servlet.

6.4 AJAX FRAMEWORK – ZK

Η τεχνολογία Asynchronous JavaScript and XML (Ajax) αποτέλεσε τον πυρήνα για την δημιουργία μιας εντελώς νέας κατηγορίας πλαισίων ανάπτυξης (Frameworks) εφαρμογών ιστού.

Το ZK[35] το οποίο χρησιμοποιείται στην πλατφόρμα «*wb*GraphFuzzyOnto», είναι ένα open-source AJAX Framework γραμμένο σε Java, το οποίο σου παρέχει την δυνατότητα να δημιουργήσεις «Rich Internet Applications» (RIA)[36], δηλαδή εφαρμογές ιστού με χαρακτηριστικά Desktop εφαρμογών, χωρίς να χρησιμοποιήσεις καθόλου JavaScript κώδικα. Αυτή η κατηγορία frameworks ονομάζεται «Indirect Ajax Frameworks» εξαιτίας αυτής της ιδιότητας τους.

Αλλά υπάρχοντα Ajax Frameworks όπως το ExtJs, το Adobe Flex Framework και το Dojo παρέχουν συγκεκριμένες βιβλιοθήκες JavaScript για την δημιουργία εφαρμογών Ιστού. Σε αντίθεση το ZK παρέχει την δυνατότητα σε ένα προγραμματιστή να υλοποιήσει μία επαφή χρησιμοποιώντας έναν μετα-ορισμό (meta-definition) που βασίζεται σε XML σε συνδυασμό με Java κλάσεις. Στην συνέχεια ο κώδικας που έχει ορίσει ο προγραμματιστής περνάει από μία φάση μεταγλώττισης κατά την οποία δημιουργούνται html σελίδες ενσωματώνοντας τεχνολογία JavaScript.

Αποτελεί μια πολύ αξιόπιστη λύση και χρησιμοποιείται ευρέως παγκοσμίως για την υλοποίηση «Client-Server» εφαρμογών με υψηλές απαιτήσεις. Το ZK εξαρτάται εξολοκλήρου από την γλώσσα Java (και κατά συνέπεια από την πλατφόρμα J2EE), όχι μόνο κατά την ανάπτυξή εφαρμογών, αλλά και για την εκτέλεση τους. Είναι στενά συνδεδεμένο με τις Servlet κλάσεις και εκτελείται σε ένα διακομιστή σε περιβάλλον Servlet Container. Παρέχει ένα μεγάλο σύνολο αντικειμένων για την σχεδίαση διεπαφών που ονομάζονται Components και τα οποία ενσωματώνουν Ajax δυνατότητες.

6.5 XHTML

Η XHTML (eXtensible HyperText Markup Language)[27] είναι μία γλώσσα σήμανσης υπερκειμένου για το διαδίκτυο και αποτελεί επίσημη πρόταση της W3C. Χαρακτηρίζεται από την μεγάλη ομοιότητας της με την HTML (και συγκεκριμένα με την HTML 4.0.1), αλλά έχει σχεδιαστεί με τέτοιο τρόπο ώστε να ώστε να συνάδει και να λειτουργεί στα πρότυπα της XML. Αυτό της δίνει την δυνατότητα και την ευχέρεια να είναι σε θέση να συνεργάζεται αρμονικά με άλλες γλώσσες και άλλους τύπους εγγράφων.

Στην περίπτωση μας προτιμήθηκε από την HTML εξαιτίας της δυνατότητας και της συμβατότητας που μας παρέχει, να ενσωματώνει xml κώδικα αρχείων SVG.

6.6 SVG

Η SVG (Scalable Vector Graphics)[28] είναι μία γλώσσα που ακολουθεί τα πρότυπα της XML όσο αφόρα την σύνταξή και την περιγραφή και χρησιμοποιείται για την δημιουργία δισδιάστατων διανυσματικών γραφικών. Τα γραφικά που παράγονται μπορεί να είναι στατικά αλλά και δυναμικά, δηλαδή να επιτρέπουν μια συγκεκριμένη αλληλεπίδραση με τον χρήστη.

Ένα SVG αρχείο ουσιαστικά είναι ένα αρχείο XML στο οποίο επιτρέπεται η ενσωμάτωση μίας γλώσσας Scripting, όπως η JavaScript μέσω της οποίας επιτρέπεται η αλληλεπίδραση με τον χρήστη και κατά συνέπεια η διαχείριση των επιλογών του χρήστη.

Οι περισσότεροι περιηγητές (browsers) υποστηρίζουν αρχεία SVG για την απεικόνιση δισδιάστατων γραφικών, με μοναδική εξαίρεση τον Internet Explorer. Αυτό βέβαια αναμένεται σύντομα να αλλάξει με την έλευση της έκδοσης 9 του Internet Explorer. Ένα SVG αρχείο υποστηρίζει συγκεκριμένα τρείς τύπους γραφικών αντικειμένων και είναι οι εξής:

- Διανυσματικά γραφικά
- Γραφικά raster
- Κείμενο

Στην «*wb*GraphFuzzyOnto» πλατφόρμα χρησιμοποιούνται SVG αρχεία για την απεικόνιση RDF γράφων, τα οποία ενσωματώνουν και την κατάλληλη λειτουργικότητα για την διαχείριση των ενεργειών του χρήστη.

6.7 CSS

Τα CSS(cascade style sheet)[30] αρχεία είναι μία τεχνική με την οποία ο προγραμματιστής μπορεί να ελέγχει τα γνωρίσματα διαφόρων αντικειμένων της σελίδας μέσα από ένα αρχείο το οποίο ενσωματώνεται σε όλες τις σελίδες. Έτσι η εμφάνιση των σελίδων μπορεί εύκολα να αλλάξει, αλλάζοντας μόνο αυτό το αρχείο.

Τα CSS (Cascading Style Sheets), είναι ένα σύνολο από ορίσματα τα οποία αναπτύχθηκαν με στόχο την καλύτερη διαχείριση της εμφάνισης των ιστοσελίδων. Μπορούμε μέσα από τα CSS, να ορίσουμε γραμματοσειρές, χρώματα, στοίχιση, backgrounds, κ.λ.π.

Στην «*wb*GraphFuzzyOnto» πλατφόρμα χρησιμοποιήθηκαν CSS κλάσεις για την καλύτερη και πιο φιλική εμφάνιση των διεπαφών του χρήστη.

6.8 JENA FRAMEWORK

Το Jena Framework[31] αποτελεί μια προγραμματιστική διεπαφή (API) σε Java, είναι ανοιχτού κώδικα και κατά κύριο λόγο παρέχει υποστήριξη για τη διαχείριση οντολογιών και προγραμματισμό εφαρμογών Σημασιολογικού Ιστού. Αναπτύσσεται κατά κύριο λόγο από τα Hewlett Packard Labs. Η πρώτη έκδοση της Jena ήταν διαθέσιμη το 2000. Η αρχιτεκτονική της επανασχεδιάστηκε και η Jena2 έγινε διαθέσιμη τον Αύγουστο του 2003, αποτελώντας πλέον την πρακτικότερη λύση για προγραμματιστική διαχείριση οντολογιών.

Μέσω της διεπαφής του Jena Framework πραγματοποιήθηκε από την πλατφόρμα «*wb*GraphFuzzyOnto», η διαχείριση των οντολογιών που είναι υλοποιημένες σε rdf γλώσσα και στην συνέχεια απεικονίζονται με rdf γράφους.

6.9 JFREECHART

Το JFreeChart[32] αποτελεί μια προγραμματιστική διεπαφή (API) σε Java, είναι ανοιχτού κώδικα και χρησιμοποιείται για την δημιουργία υψηλής ποιότητας γραφημάτων τόσο σε μορφές αρχείων εικόνας (π.χ. .jpeg, .png) όσο και σε μορφές διανυσματικών γραφικών (π.χ. .eps, .svg, .pdf).

Μέσω του JFreeChart δημιουργούμε τα απαραίτητα επιτραπέζια γραφήματα που απαιτούνται για την ασαφοποίηση των Ασαφών Τύπων Δεδομένων (Fuzzy DataTypes) , όπως αναφέραμε στο κεφάλαιο 4.3.2.

6.10 DOT

Η γλώσσα προγραμματισμού DOT[33] είναι μια περιγραφική γλώσσα για την δημιουργία γράφων. Ένα αρχείο τύπου DOT μπορεί εύκολα να επεξεργαστεί σε έναν επεξεργαστή κείμενου. Με την γλώσσα προγραμματισμού DOT παράγονται γράφοι όπως για παράδειγμα ένας rdf γράφος σε διάφορες μορφές αρχείων, όπως SVG, PNG, GIF και PostScript. Για την απεικόνιση και την δημιουργία ενός γράφου από ένα αρχείο DOT απαιτείται η εγκατάσταση του προγράμματος GraphVisualization. Οι οδηγίες για την εγκατάσταση του περιγράφονται στο κεφάλαιο 12.2.

7 ΜΟΝΑΔΕΣ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ

Το παρόν κεφάλαιο επιχειρεί μία εις βάθος ανάλυση των δομικών στοιχείων που απαρτίζουν την δικτυακή πλατφόρμα «*wb*GraphFuzzyOnto». Η σχεδίαση των διεπαφών όπως έχουμε ήδη αναφέρει, πραγματοποιήθηκε κάνοντας χρήση του Ajax Framework ZK και συγκεκριμένα με τα «ZK Components» που το συγκεκριμένο Ajax framework παρέχει.

Για την καλύτερη κατανόηση του αναγνώστη, κρίθηκε σκόπιμο να πραγματοποιηθεί αρχικά μία περιγραφή των ZK Components που έχουν χρησιμοποιηθεί στον σχεδιασμό της διεπαφής της πλατφόρμας ασαφοποίησης και στην συνέχεια θα αναφερθούμε στις Java κλάσεις και στις JavaScript συναρτήσεις που παρέχουν όλη την επιθυμητή λειτουργικότητα της πλατφόρμας ασαφοποίησης οντολογιών.

7.1 ZK COMPONENTS

Μία ZK σελίδα αποτελείται από μία ιεραρχία ετικετών XML, όπου η κάθε μία αντιστοιχεί σε μία Java κλάση που αντιστοιχεί σε ένα συγκεκριμένο component. Για παράδειγμα η ετικέτα **<window id="mainWindow"></window** αναφέρεται στην κλάση **org.zkoss.zul.Window**. Επίσης οι ετικέτες μπορούν να περιέχουν αναφορές σε μεθόδους για τον χειρισμό συγκεκριμένων ενεργειών, όπως για παράδειγμα το πάτημα ενός κουμπιού ή επιλογές που σχετίζονται με την εμφάνιση των components.

Τα ZK Components αντιστοιχούν σε ενισχυμένες διαδραστικές HTML δομές και έχουν διπλή υπόσταση. Αφενός είναι αντικείμενα μίας Java κλάσης και αφετέρου αντιστοιχούν σε αντικείμενα του DOM (Μοντέλο Δυναμικών Αντικείμενων)[37] δέντρου της σελίδας. Ο μηχανισμός του ZK εξασφαλίζει ότι οι δύο αυτές υποστάσεις έχουν πάντα συγχρονισμένες καταστάσεις, δηλαδή αν πραγματοποιήσουμε μία αλλαγή σε ένα συγκεκριμένο αντικείμενο μιας JAVA κλάσης ενός component, αυτή θα αντικατοπτρίζεται στο αντίστοιχο αντικείμενο της σελίδας ZK στο οποίο αναφέρεται και αν προκληθεί για παράδειγμα κάποιο γεγονός από ένα αντικείμενο μιας ZK σελίδας, αυτό εντοπίζεται από το αντίστοιχο αντικείμενο της Java κλάσης.

Στην συνεχεία περιγράφουμε τα βασικά components που χρησιμοποιήθηκαν για τον σχεδιασμό της διεπαφής της πλατφόρμας ασαφοποίησης ανά κατηγορία στην οποία ανήκουν.

- Δομικά στοιχεία της σελίδας που αντιστοιχούν σε περιοχές της σελίδας και έχουν την δυνατότητα να περικλείουν αλλά components. Σε αυτήν την κατηγορία ανήκουν τα εξής στοιχεία:
    - **Window**. Είναι ένα component παραθύρου το οποίο μπορεί να περικλείει διάφορα άλλα components ZK.

- **Div**. Είναι ένα component ορισμού μιας περιοχής σε μία σελίδα, παρόμοιο με στοιχείο <Div> που υπάρχει σε μία σελίδα HTML.
- **HBOX**. Είναι ένα component ορισμού μιας περιοχής σε μία σελίδα, με την μοναδική διαφορά με ένα component DIV να έχει να κάνει ότι τα components που περικλείονται μέσα σε ένα HBOX τοποθετούνται σε οριζόντια διάταξη.
- **VBOX**. Είναι ένα component ορισμού μιας περιοχής σε μία σελίδα, με την μοναδική διαφορά με ένα component DIV να έχει να κάνει ότι τα components που περικλείονται μέσα σε ένα VBOX τοποθετούνται σε κάθετη διάταξη.
- **Separator**. Είναι ένα component που παρέχει την ιδία λειτουργικότητα με ένα στοιχείο
 της HTML με τη διαφορά ότι μπορείς να ορίσεις επιπλέον ιδιότητες όπως για παράδειγμα height.
- **Tabbox**. Πρόκειται για ένα component που σου παρέχει την δυνατότητα ορισμού διακριτών περιοχών με την μορφή «Tabs», όπως για παράδειγμα τα tabs που υπάρχουν στους περιηγητές (browsers) του διαδικτύου. Ένα component τύπου Tabbox θα πρέπει να περικλείει οπωσδήποτε τα components **tab** και **tabpanel**.

- Στοιχεία φόρμας και κειμένου. Σε αυτήν την κατηγορία ανήκουν τα εξής στοιχεία:
  - **Textbox**. Είναι ένα component που περεχεί την δυνατότητα εισαγωγής κείμενου.
  - **Doublebox**. Είναι ένα component που περεχεί την δυνατότητα εισαγωγής δεκαδικών αριθμών.
  - **Button.** Είναι ένα component τύπου κουμπιού όπως αυτά που συναντάμε σε μία φόρμα στοιχείων HTML.
  - **Label.** Είναι ένα component που παρέχει παρόμοια λειτουργικότητα με ένα στοιχείο <label> της HTML.
  - **Listbox.** Είναι ένα component τύπου Λίστας το οποία παρέχει πολύ περισσότερες δυνατότητες από το αντίστοιχο στοιχείο της HTML τόσο σε θέματα διάταξης του περιεχομένου αλλά και των δυνατοτήτων που ενσωματώνει.
  - **Paging.** Είναι ένα component το οποία χρησιμοποιείται σε συνδυασμό με ένα component τύπου Listbox και όχι μόνο, για την σελιδοποίηση των εγγραφών που περιέχονται σε μία λίστα ορίζοντας του ένα συγκεκριμένο αριθμό εγγραφών που θα πρέπει να εμφανίζονται σε κάθε «σελίδα» της λίστας.

- Στοιχεία εργαλειοθήκης και μενού πλοήγησης. Σε αυτήν την κατηγορία ανήκουν τα εξής στοιχεία:
  - **Menupopup**. Πρόκειται για ένα αναδυόμενο μενού πλοήγησης όπως αυτά υπάρχουν σε όλες τις desktop εφαρμογές. Το μενού πλοήγησης αναδύεται με το που ο χρήστης πατήσει το δεξί κλικ του ποντικιού πάνω στο στοιχείο που έχει ορισθεί να έχει αναδυόμενο μενού. Ένα component τύπου menupopup θα πρέπει να περικλείει οπωσδήποτε component τύπου **Menuitem**.
- Στοιχεία Html. Σε αυτήν την κατηγορία ανήκουν τα εξής στοιχεία:
  - **Html**. Ένα αντικείμενο τύπου HTML, παρέχει την δυνατότητα εισαγωγής κώδικα HTML.
  - **Iframe**. Ένα αντικείμενο τύπου IFRAME, παρέχει την ίδια λειτουργικότητα με ένα στοιχείο <iframe> της HTML.
- Στοιχεία γραφημάτων. Σε αυτήν την κατηγορία ανήκει το στοιχείο:
  - **Chart**. Κάνοντας χρήση των αντικείμενων του συγκεκριμένου component μπορούμε να προβούμε στην δημιουργία γραφημάτων. Το component chart βασίζεται στις βιβλιοθήκες του JFreeChart για την κατασκευή των γραφημάτων.

7.2 Περιγραφη Διεπαφησ Χρηστων

Η σελίδα «index.zul» αποτελεί την κύρια διεπαφή των χρηστών μέσω της οποία επιτυγχάνεται η ασαφοποίηση μίας οντολογίας. Η συγκεκριμένη σελίδα έχει υλοποιηθεί στο σύνολό της από αντικείμενα των ZK Components που αναφέραμε παραπάνω. Στην συνέχεια εμφανίζεται ο υλοποιημένος κώδικας της index.zul σελίδας μαζί με τις απαραίτητες διευκρινήσεις και επισημάνσεις για την καλύτερη κατανόηση του.

Index.zul

```
<?xml version="1.0" encoding="UTF-8"?>
<?page id="MainPage"?>

<window id="mainWindow"
    xmlns="http://www.zkoss.org/2005/zul"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.zkoss.org/2005/zul http://www.zkoss.org/2005/zul/zul.xsd"
    use="zul.ConstMainWin">
```

Επεξήγηση

Ορίζουμε αρχικά ένα αντικείμενο τύπου Window που θα περικλείει όλα τα αντικείμενα τα οποία επιθυμούμε. Με την ιδιότητα ***use*** ορίζουμε την Java κλάση μέσα από την οποία θα μπορούμε να διαχειριζόμαστε τα components και τις αποκρίσεις που προκύπτουν από τις ενέργειες των χρηστών. Στο σύνολο της, σχεδόν όλη η λειτουργικότητα και η διαδραστικότητα της πλατφόρμας «*wb*GraphFuzzyOnto» πραγματοποιείται μέσα από τους μεθόδους της Java κλάσης που έχουμε ορίσει. Στην συνέχεια ορίζουνε ένα αντικείμενο τύπου tabbox το οποίο θα αποτελείται από πέντε αντικείμενα τύπου tabs και πέντε αντικείμενα τύπου tabpanel.

```
<div align="center" style="background-color:whitesmoke;">
    <label value="wbGraphFuzzyOnto" sclass="lbl"/>
    <tabbox id="tb" orient="vertical" width="1000px" height="1000px" style="background-
color:white;">
      <tabs width="100px">
        <tab label="Input Text" selected="true" id="tab1"/>
        <tab label="Step1" id="tabstep1" visible="false"/>
        <tab label="Step2a" id="tabstep2a" visible="false"/>
        <tab label="Step2b" id="tabstep2b" visible="false"/>
        <tab label="Step2c" id="tabstep2c" visible="false"/>
      </tabs>
      <tabpanels>
       <tabpanel id="step0panel">
          <div align="center">
            <vbox>
              <label value="Input Text" sclass="lbl"/>
              <textbox width="600px" height="600px" cols="70" rows="10" id="rdfText"
sclass="atextbox"/>
              <separator/>
              <hbox align="center">
                <button label="Clear Text" id="clearBText"
onClick="mainWindow.onClearBtextCkick();"/>
                <button label="Parse Text" id="parseBText"
onClick="mainWindow.onParseBtextCkick();"/>
              </hbox>
            </vbox>
          </div>
        </tabpanel>
```

Επεξήγηση

Το step0panel παρέχει την κατάλληλη λειτουργικότητα για την εισαγωγή του κώδικα της crisp οντολογίας. Το **αντικείμενο rdfText τύπου textbox** χρησιμοποιείται για την εισαγωγή της crisp οντολογίας που είναι υλοποιημένη σε RDF σημασιολογική γλώσσα**.** Το **αντικείμενο clearBText τύπου button** χρησιμοποιείται για την εκκαθάριση του περιεχόμενου του αντικειμένου **rdfText.** Το **αντικείμενο parseBText τύπου button** προκαλεί την δημιουργία του RDF γράφου της crisp οντολογίας.

```
<tabpanel id="step1panel">
  <div align="left">
    <vbox>
      <label value="Visualize Ontology" sclass="lbl"/>
      <separator/>
      <hbox style="margin-left:1000px;">
        <button id="exportButton" image="/images/img/collapse_d.gif"
            hoverImage="/images/img/collapse_e.gif"
            tooltiptext="Export to XML" style="margn-right:20px;cursor:pointer;"
            disabled="true"
            onClick="mainWindow.exportStep1(1);"/>
        <button id="nextButton" image="/images/img/next1_d.png"
            hoverImage="/images/img/next1_e.png" tooltiptext="Next Step" disabled="true"
            style="cursor:pointer;" onClick="mainWindow.goTostep2();"/>
      </hbox>
      <separator/>
      <div id="iframeDiv" width="100%" height="100%">
        <iframe id="iframe" width="1120px" height="950px"/>
      </div>
    </vbox>
  </div>
</tabpanel>
```

Επεξήγηση

Το step1panel ενσωματώνει τον RDF γράφο και παρέχει την κατάλληλη λειτουργικότητα για την επιλογή των ασαφών στοιχείων μιας οντολογίας. Το **αντικείμενο exportButton τύπου button** χρησιμοποιείται για την εξαγωγή σε xml κώδικα όλων των επιλογών που έχει πραγματοποιήσει ο χρήστης. Το **αντικείμενο iframeDiv τύπου DIV** περιλαμβάνει τον RDF γράφο**. Το αντικείμενο nextButton τύπου button** επιτρέπει την δυνατότητας μεταφοράς σε άλλο tab, ανάλογα με τις επιλογές που έχει πραγματοποιήσει ο χρήστης.

```
        <tabpanel id="step2apanel">
          <div align="left">
            <vbox>
              <label value="Visualize Ontology" sclass="lbl"/>
              <separator/>
              <hbox style="margin-left:1000px;">
                <button                                          image="/images/img/collapse_d.gif"
hoverImage="/images/img/collapse_e.gif"
                    tooltiptext="Export to XML" style="margn-right:20px;cursor:pointer;"
                    onClick="mainWindow.exportStep2a(1);"/>
                <button image="/images/img/next1_d.png"
                    hoverImage="/images/img/next1_e.png" tooltiptext="Next Step"
                    style="cursor:pointer;" onClick="mainWindow.goTostep2b();"/>
              </hbox>
              <hbox>
                <iframe id="step2aFrame" width="750px" height="1100px" align="center"/>
                <vbox align="left" style="margin-top:15px">
                  <listbox fixedLayout="true" width="400px" id="relationListbox">
                    <auxhead>
                      <auxheader label="Relations" colspan="3" align="center"/>
                    </auxhead>
                    <listhead sizable="true">
                      <listheader label="Relation" sort="auto" align="left"/>
                      <listheader label="Vagueness Nature" sort="auto" align="left"/>
                      <listheader label="Degree Interpretaion" sort="auto" align="left"/>
                    </listhead>
                  </listbox>
                  <separator/>
                  <listbox fixedLayout="true" width="400px" id="relationStatement">
                    <auxhead>
                      <auxheader label="Statements" colspan="2" align="center"
                            style="background-color:LightSteelBlue;height:17px;color:window;"/>
                    </auxhead>
                    <listhead sizable="false">
                      <listheader label="Relation Statement" sort="none" align="left"
                             width="300px"/>
                      <listheader label="Fuzzy Degree" sort="none" align="left" width="100px"/>
                    </listhead>
                  </listbox>
                </vbox>
```

```
            </hbox>
          </vbox>
        </div>
      </tabpanel>
```

Επεξήγηση

Το step2apanel ενσωματώνει τον RDF γράφο επισημαίνοντας τις ασαφείς σχέσεις που έχει επιλέξει ο χρήστης σε προηγούμενο βήμα. Παρέχει το κατάλληλο μηχανισμό για την συμπλήρωση των στοιχείων του Degree Interpretation και Vague Nature για κάθε ασαφή σχέση καθώς και του βαθμού ασάφειας μεταξύ των εμπλεκόμενων μερών (στιγμιότυπων κλάσεων). Υπάρχουν δύο αντικείμενα **τύπου button,** που χρησιμοποιούνται για την εξαγωγή σε xml κώδικα όλων των επιλογών που έχει πραγματοποιήσει ο χρήστης και για την δυνατότητα μεταφοράς σε άλλο tab ανάλογα με τις επιλογές που έχει πραγματοποιήσει ο χρήστης. Το **αντικείμενο iframeDiv τύπου DIV** περιλαμβάνει τον RDF γράφο. Τα αντικείμενα τύπου Listbox, **relationListbox** και **relationStatement** χρησιμοποιούνται για την απόδοση των στοιχείων του Degree Interpretation και Vague Nature για κάθε ασαφή σχέση καθώς και του βαθμού ασάφειας μεταξύ των εμπλεκόμενων μερών.

```
<tabpanel id="step2bpanel">
      <div align="left">
        <vbox>
          <label value="Visualize Ontology" sclass="lbl"/>
          <separator/>
          <hbox style="margin-left:1000px;">
            <button                                    image="/images/img/collapse_d.gif"
hoverImage="/images/img/collapse_e.gif"
                tooltiptext="Export to XML" style="margn-right:20px;cursor:pointer;"
                onClick="mainWindow.exportStep2b(1);"/>
            <button image="/images/img/next1_d.png"
                hoverImage="/images/img/next1_e.png" tooltiptext="Next Step"
                style="cursor:pointer;" onClick="mainWindow.goTostep2c();"/>
          </hbox>
          <hbox>
            <iframe id="step2bFrame" width="750px" height="1100px" align="center"/>
            <vbox align="left" style="margin-top:15px">
              <listbox fixedLayout="true" width="400px" id="conceptListbox">
                <auxhead>
                  <auxheader label="Concepts" colspan="3" align="center"/>
                </auxhead>
                <listhead sizable="true">
                  <listheader label="Concept" sort="auto" align="left"/>
                  <listheader label="Vagueness Nature" sort="auto" align="left"/>
                  <listheader label="Degree Interpretaion" sort="auto" align="left"/>
                </listhead>

              </listbox>
```

```
                    <separator/>
                    <listbox fixedLayout="true" width="400px" id="conceptInstances">
                      <auxhead>
                        <auxheader label="Concept Instances" colspan="2" align="center"
                              style="background-color:LightSteelBlue;height:17px;color:window;"/>
                      </auxhead>
                      <listhead sizable="false">
                        <listheader label="Concept Instance" sort="none" align="left"
                              width="300px"/>
                        <listheader label="Fuzzy Degree" sort="none" align="left" width="100px"/>
                      </listhead>
                    </listbox>
                  </vbox>
                </hbox>
              </vbox>
            </div>
          </tabpanel>
```

Επεξήγηση

Το step2bpanel ενσωματώνει τον RDF γράφο επισημαίνοντας τις ασαφείς έννοιες που έχει επιλέξει ο χρήστης σε προηγούμενο βήμα. Παρέχει το κατάλληλο μηχανισμό για την συμπλήρωση των στοιχείων του Degree Interpretation και Vague Nature για κάθε ασαφή έννοια καθώς και του βαθμού ασάφειας μεταξύ των εμπλεκόμενων μερών (δηλαδή το κατά πόσο ένα στιγμιότυπο ανήκει σε αυτήν την έννοια). Υπάρχουν δύο αντικείμενα **τύπου button,** που χρησιμοποιούνται για την εξαγωγή σε xml κώδικα όλων των επιλογών που έχει πραγματοποιήσει ο χρήστης και για την δυνατότητα μεταφοράς σε άλλο tab ανάλογα με τις επιλογές που έχει πραγματοποιήσει ο χρήστης. Το **αντικείμενο iframeDiv τύπου DIV** περιλαμβάνει τον RDF γράφο. Τα αντικείμενα τύπου Listbox, **conceptListbox** και **conceptInstances** χρησιμοποιούνται για την απόδοση των στοιχείων του Degree Interpretation και Vague Nature για κάθε ασαφή εννοια καθώς και του βαθμού ασάφειας.

```
          <tabpanel id="step2cpanel">
            <div align="left">
              <vbox>
                <label value="Visualize Ontology" sclass="lbl"/>
                <separator/>
                <hbox style="margin-left:1000px;">
                  <button                                    image="/images/img/collapse_d.gif"
hoverImage="/images/img/collapse_e.gif"
                      tooltiptext="Export to XML" style="margn-right:20px;cursor:pointer;"
                      onClick="mainWindow.exportStep2c(1);"/>
                </hbox>
                <hbox>
                  <iframe id="step2cFrame" width="750px" height="1100px" align="left"/>
                  <vbox align="left" style="margin-top:15px">
                    <label id="attributelLbl" sclass="lbl1"/>
```

```
<html id="htmlCode"/>
<listbox fixedLayout="true" width="510px" id="fuzzyDatatypes" mold="paging">
  <auxhead>
    <auxheader label="Fuzzy Datatypes" colspan="9" align="center"/>
  </auxhead>
  <auxhead>
    <auxheader label="Attributes-Datatypes" colspan="1" align="center"
        width="230px"/>
    <auxheader label="Values" colspan="8" align="center"/>
  </auxhead>
  <auxhead>
    <auxheader label="Attributes-Datatypes" colspan="1" align="left"/>
    <auxheader label="A" colspan="2" align="center"/>
    <auxheader label="B" colspan="2" align="center"/>
    <auxheader label="C" colspan="2" align="center"/>
    <auxheader label="D" colspan="2" align="center"/>
  </auxhead>
  <listhead>
    <listheader label="Attributes-Datatypes" align="left" width="230px"/>
    <listheader label="xA" sort="none" align="left" width="35px"/>
    <listheader label="yA" sort="none" align="left" width="35px"/>
    <listheader label="xB" sort="none" align="left" width="35px"/>
    <listheader label="yB" sort="none" align="left" width="35px"/>
    <listheader label="xC" sort="none" align="left" width="35px"/>
    <listheader label="yC" sort="none" align="left" width="35px"/>
    <listheader label="xD" sort="none" align="left" width="35px"/>
    <listheader label="yD" sort="none" align="left" width="35px"/>
  </listhead>
</listbox>
<paging id="datatypesPaging" pageSize="20" autohide="false" detailed="true"/>
<zscript>fuzzyDatatypes.paginal = datatypesPaging;</zscript>

<menupopup id="editPopup">
  <menuitem image="/images/img/Centigrade-Widget-Icons/kchart1.png"
      label="Export Chart only for this Datatype"
      onClick="mainWindow.createChart(1);"/>
  <menuseparator/>
  <menuitem image="/images/img/Centigrade-Widget-Icons/kchart.png"
      label="Export Chart for this Attribute with all Datatypes"
      onClick="mainWindow.createChart(2);"/>
</menupopup>
</vbox>
```

```
                </hbox>
              </vbox>
            </div>

                  </tabpanel>
```

Επεξήγηση

Το step2cpanel ενσωματώνει τον RDF γράφο επισημαίνοντας τις ασαφείς ιδιότητες που έχει επιλέξει ο χρήστης σε προηγούμενο βήμα, σε συνδυασμό με τα λεκτικά στα οποία αναφέρονται και αποδίδουν ασαφή γνώση. Παρέχει το κατάλληλο μηχανισμό για την συμπλήρωση του πεδίου τιμών για κάθε λεκτικό, βάση του γραφικού διαγράμματος τον οποίο τον καθοδηγεί. Υπάρχουν ένα αντικείμενο **τύπου button,** που χρησιμοποιείται για την εξαγωγή σε xml κώδικα όλων των επιλογών που έχει πραγματοποιήσει ο χρήστης. Το **αντικείμενο iframeDiv τύπου DIV** περιλαμβάνει τον RDF γράφο. Το αντικείμενο τύπου Listbox, **fuzzyDatatypes** χρησιμοποιείται για την απόδοση του πεδίου ορισμού για κάθε λεκτικό που αναφέρεται σε μια ιδιότητα. Το αντικείμενο τύπου **Paging** χρησιμοποιείται για την σελιδοποίηση των εγγραφών του αντικείμενου **fuzzyDatatypes.** Το αντικείμενο **editPopup** τύπου Menopopup χρησιμοποιείται για να παρέχει στον χρήστη επιλογές εμφάνισης γραφικών διαγραμμάτων του πεδίου τιμών των λεκτικών που αποδίδουν ασαφή γνώση.

```
      </tabpanels>
    </tabbox>
  </div>
  <window id="chartWindow" width="600px" height="600px" border="normal"
      action="onshow:anima.appear(#{self});onhide:anima.fade(#{self})" visible="false">
    <caption id="capLbl"/>
    <separator/>
    <vbox>
      <div align="right">
        <button style="cursor:pointer;" image="/images/img/Centigrade-Widget-Icons/Exit.png"
            onClick="hideWindow();" tooltiptext="Close"/>
      </div>
      <zscript>
        void hideWindow(){
        chartWindow.setVisible(false);
        }
      </zscript>
      <chart id="step" width="500" height="500"
          type="area" threeD="true" fgAlpha="128"/>
    </vbox>
  </window>
</window>
```

Το αντικείμενο **chartWindow τύπου Window** χρησιμοποιείται για την εμφάνιση των γραφικών διαγραμμάτων που θα αναπαριστούν τα καθορισμένα όρια ενός λεκτικού που αποτελεί τιμή σε μία ιδιότητα ή για την συνολική αναπαράσταση των λεκτικών που αποτελούν τιμές σε μία συγκεκριμένη ιδιότητα. Το **αντικείμενο step τύπου Chart** χρησιμοποιείται για την δημιουργία των γραφικών διαγραμμάτων, η ιδιότητα type ορίζει το είδος του γραφήματος, η ιδιότητα threeD αν είναι αληθής επιτρέπει την εμφάνιση με τρισδιάστατη μορφή, ενώ η τιμή της ιδίοτητας **fgAlpha** ορίζει την διαφάνεια των τμημάτων του γραφικού διαγράμματος.

7.3 ΚΛΑΣΕΙΣ JAVA

Στην ενότητα αυτή θα παρουσιαστούν οι Java κλάσεις της πλατφόρμας «*wb*GraphFuzzyOnto», ο ρόλος τους και τα σημαντικότερα σημεία τους. Πριν ξεκινήσουμε την περιγραφή όλων των κλάσεων, κρίνεται σκόπιμο η αναφορά ορισμένων επισημάνσεων. Οι Java κλάσεις που έχουν υλοποιηθεί είναι οι ακόλουθες:

- Package: servlets
  - Step1Servlet.java (Servlet)
  - Step2aServlet.java (Servlet)
  - Step2bServlet.java (Servlet)
  - Step2cServlet.java (Servlet)
  - XmlServlet.java (Servlet)
- Package: zul.admin_components
  - Step1AdminComponents.java
  - Step2aAdminComponents.java
  - Step2bAdminComponents.java
  - Step2cAdminComponents.java
- Package: zul.update
  - Functions.java
  - UpdateSvgFileStep1.java
  - UpdateSvgFileStep2a.java
  - UpdateSvgFileStep2b.java
  - UpdateSvgFileStep2c.java

- Package: graph
    - OntoVisualize.java
        - ✓ SH (Εμφωλευμένη Κλάση)
        - ✓ SaxErrorHandler (Εμφωλευμένη Κλάση)
- Package: zul
    - ConstMainWin.java
- Package: zul.onload
    - SetSvgSelections.java

Σημαντικό, κρίνεται να αναφέρουμε ότι κατά την διαδικασία ανάπτυξης της πλατφόρμας παρουσιαστήκαν κάποια προβλήματα ως προς τον τρόπο που έπρεπε να διαχειριζόμαστε τις επιλογές του χρήστη που πραγματοποιούσε πάνω σε έναν RDF γράφο. Όπως έχουμε ήδη αναφέρει για την απεικόνισή του γράφου χρησιμοποιήσαμε SVG αρχεία ενσωματωμένα σε XHTML σελίδες, σε συνδυασμό με JavaScript συναρτήσεις που επιτρέπουν την αλληλεπίδραση με τον χρήστη. Επειδή δεν είναι δυνατόν να διαχειριστούμε απευθείας ZK components μέσα από JavaScript συναρτήσεις, αποφασίστηκε να χρησιμοποιούμε Java Servlet κλάσεις που μπορούν να κληθούν μέσα από JavaScript συναρτήσεις, μέσω AJAX μεθόδων. Την επίτευξη της διαχείρισης των ZK components ολοκληρώνεται μέσα από τους μεθόδους που παρέχουν οι κλάσεις του πακέτου zul.admin_components όπως θα περιγράψουμε στην συνέχεια.

Επίσης θα πρέπει να επισημάνουμε ότι το βήμα «step1» αντιστοιχεί στην επιλογή των στοιχείων της crisp οντολογίας που αποδίδουν ασάφεια (το οποίο επιτυγχάνεται στο tab «tabstep1» και tabpanel «step1panel»), το βήμα «step2a» αντιστοιχεί στην ασαφοποίηση των ασαφών σχέσεων που έχουνε επιλεχτεί (το οποίο επιτυγχάνεται στο tab «tabstep2a» και tabpanel «step2apanel»), το βήμα «step2b» αντιστοιχεί στην ασαφοποίηση των ασαφών εννοιών που έχουνε επιλεχτεί (το οποίο επιτυγχάνεται στο tab «tabstep2b» και tabpanel «step2bpanel») και τέλος το βήμα «step2c» αντιστοιχεί στην ασαφοποίηση των ασαφών λεκτικών που αποτελούν τιμές μιας ιδιότητες (το οποίο επιτυγχάνεται στο tab «tabstep2c και tabpanel «step2cpanel»).

7.3.1 *CONSTMAINWIN.JAVA*

Περιγραφή Λειτουργίας:

Η ConstMainWin.java κλάση έχει οριστεί στην index.zul σελίδα για την διαχείριση των ZK components της συγκεκριμένης σελίδας. Μέσα από την λειτουργικότητα που παρέχει η συγκεκριμένη κλάση, δημιουργούνται νέα components που αναφέρονται στην index.zul σελίδα, διαχειρίζονται όλες τις αιτήσεις και τις αποκρίσεις των χρηστών, γίνονται

κλήσεις σε μεθόδους άλλων κλάσεων. Ουσιαστικά πρόκειται για την κλάση, μέσα από την εκτέλεση των μεθόδων της, επιτυγχάνεται η λειτουργικότητα της πλατφόρμας ασαφοποίησης οντολογιών «*wb*GraphFuzzyOnto».

Κλήσεις:

```
import org.zkoss.zul.*;
import org.zkoss.zul.Button;
import org.zkoss.zul.Window;
import org.zkoss.zk.ui.Desktop;
import org.zkoss.zk.ui.Executions;
import org.zkoss.zk.ui.event.Events;
import org.zkoss.zk.ui.event.EventListener;
import org.zkoss.zk.ui.event.Event;
import org.xml.sax.InputSource;
import org.w3c.dom.*;
import java.util.*;
import java.io.StringReader;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.File;
import com.hp.hpl.jena.rdf.model.*;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import zul.admin_components.Step1AdminComponents;
import zul.admin_components.Step2aAdminComponents;
import zul.admin_components.Step2bAdminComponents;
import zul.admin_components.Step2cAdminComponents;
import zul.update.UpdateSvgFileStep2a;
import zul.update.UpdateSvgFileStep2b;
import zul.update.UpdateSvgFileStep2c;
import graph.OntoVisualize;
```

Μεταβλητές Κλάσης:

```
Button clearBText; -> Αντιστοιχεί στο Component με id clearBText της index.zul
Button parseBText; -> Αντιστοιχεί στο Component με id parseBText της index.zul
Textbox rdfText; -> Αντιστοιχεί στο Component με id rdfText της index.zul
Textbox uriText; -> Αντιστοιχεί στο Component με id rdfText της index.zul
Tabbox tb; -> Αντιστοιχεί στο Component με id tb της index.zul
-> Αντιστοιχεί στο Component με id tabstep1 της index.zul. Αντιστοιχεί στο tab στο οποίο γίνεται η
επιλογή των ασαφών στοιχείων
public static Tab tabstep1;
-> Αντιστοιχεί στο Component με id tabstep2a της index.zul. Αντιστοιχεί στο tab στο οποίο γίνεται η
ασαφοποίηση των σχέσεων
public static Tab tabstep2a;
-> Αντιστοιχεί στο Component με id tabstep2b της index.zul. Αντιστοιχεί στο tab στο οποίο γίνεται η
ασαφοποίηση των εννοιών
public static Tab tabstep2b;
-> Αντιστοιχεί στο Component με id tabstep2c της index.zul. Αντιστοιχεί στο tab στο οποίο γίνεται η
ασαφοποίηση των λεκτικών
public static Tab tabstep2c;
Div step1Div; -> Αντιστοιχεί στο Component με id step1Div της index.zul
Iframe step1Iframe; -> Αντιστοιχεί στο Component με id step1Iframe της index.zul
Iframe step2aFrame; -> Αντιστοιχεί στο Component με id step2aFrame της index.zul
Iframe step2bFrame; -> Αντιστοιχεί στο Component με id step2bFrame της index.zul
Iframe step2cFrame; -> Αντιστοιχεί στο Component με id step2cFrame της index.zul
Html htmlCode; -> Αντιστοιχεί στο Component με id htmlCode της index.zul
```

```
Label attributeLbl; -> Αντιστοιχεί στο Component με id attributeLbl της index.zul
Menupopup editPopup; -> Αντιστοιχεί στο Component με id editPopup της index.zul
Window chartWindow; -> Αντιστοιχεί στο Component με id chartWindow της index.zul
public static Listbox relationListbox; -> Αντιστοιχεί στο Component με id relationListbox της index.zul
public static Listbox relationStatement; -> Αντιστοιχεί στο Component με id relationStatement της index.zul
public static Listbox conceptInstances; -> Αντιστοιχεί στο Component με id conceptInstances της index.zul
public static Listbox conceptListbox;  -> Αντιστοιχεί στο Component με id conceptListbox της index.zul
public static Listbox fuzzyDatatypes;  -> Αντιστοιχεί στο Component με id fuzzyDatatypes της index.zul
public static Button nextButton; -> Αντιστοιχεί στο Component με id nextButton της index.zul
public static Button exportButton; -> Αντιστοιχεί στο Component με id exportButton της index.zul
-> Μεταβλητή τύπου String που χρησιμοποιείται για την εξαγωγή του RDF κώδικα και
    των επιλογών που έχει πραγματοποιήσει ο χρήστης σε XML μορφή
public static String stepXML = "";
-> Μεταβλητή που  αποθηκεύει κάθε φόρα σε πιο βήμα - πάνελ βρίσκεται ο χρήστης  χρησιμοποιείται για
   εξαγωγή XML
public static String step;
->  Μεταβλητή τύπου Vector για την αποθήκευση των επιλογών που πραγματοποιεί
     ο χρήστης στο Tabpanel με id step1panel (Πάνελ επιλογής ασαφών στοιχείων)
public static Vector selections;
->  Το συγκεκριμένο αντικείμενο χρησιμοποιείται για να έχουμε πρόσβαση σε όλα τα components
     public static Desktop desktop;
-> Στιγμιότυπο της κλάσης Step2aAdminComponents
public static Step2aAdminComponents step2aAdminComponents;
-> Στιγμιότυπο της κλάσης Step2bAdminComponents
public static Step2bAdminComponents step2bAdminComponents;
-> Στιγμιότυπο της κλάσης Step2cAdminComponents
public static Step2cAdminComponents step2cAdminComponents;
-> Στιγμιότυπο της κλάσης Step1AdminComponents
public static Step1AdminComponents step1AdminComponents;
-> Μεταβλητή που  αποθηκεύει κάθε φόρα σε πιο βήμα - πάνελ βρίσκεται ο χρήστης
public static String val = "";
->  Αποθηκεύει τις ασαφείς έννοιες που έχει επιλέξει ο χρήστης
public static Vector vect_1 = new Vector();
->  Αποθηκεύει τις ασαφείς σχέσεις που έχει επιλέξει ο χρήστης
public static Vector vect_2 = new Vector();
->  Αποθηκεύει τις ιδιότητες και τα ασαφή λεκτικά που έχει επιλέξει ο χρήστης
public static Vector vect_3 = new Vector();
->  Directory στο οποίο δημιουργούνται και αποθηκεύονται τα svg και XHTML αρχεία
public static String tmpDir =  "C:\\ wbGraphFuzzyOnto \\out\\exploded\\TestRdfValidatorWeb\\tmp";
-> Χρησιμοποιείται για την διαχωρισμό των επίλογων που έχει πραγματοποιήσει ο χρήστης
public static HashMap xmlData = new HashMap();
-> Όνομα αρχείου SVG χωρίς κατάληξη
public static String fileNameWithoutSuffix = "";
-> Όνομα αρχείου SVG χωρίς κατάληξη
public static String svgFileName = "";
```

Μέθοδοι:

- public void onCreate()

Η συγκεκριμένη μέθοδος εκτελείται με το πού θα κληθεί η ZK σελίδα index.zul. Αρχικά αρχικοποιούνται όλες οι μεταβλητές που αναφέραμε παραπάνω και που αντιστοιχούν σε components της σελίδας index.zul. Αυτό επιτυγχάνεται για κάθε component κάνοντας χρήση της μεθόδου getFellowIfAny("αναγνωριστικό") και του αναγνωριστικού του. Επίσης στο component «tb» που είναι τύπου Tabbox εκχωρείται ένας «EventListener» για την διαχείριση των tabs όταν επιλέγεται ενά tab. Συγκεκριμένα με την

επιλογή ενός tab εκχωρούμε μία συγκεκριμένη τιμή στην μεταβλητή val που να αντιπροσωπεύει το επιλεγμένο tab. Τέλος αρχικοποιείται το αντικείμενο desktop με την μέθοδο getDesktop().

- public void createChart(int number)

Η συγκεκριμένη μέθοδος εκτελείται με το που ο χρήστης επιλέξει μία από τις διαθέσιμες επιλογές του component editpopup που του παρέχονται στην σελίδα index.zul. Αν η τιμή number ισούται με 1 τότε δημιουργείται το αντίστοιχο γραφικό διόραμα για το στοιχείο του component fuzzyDatatypes που επέλεξε ο χρήστης. Συγκεκριμένα εμφανίζεται το γραφικό διάγραμμα αναπαράστασης των καθορισμένων ορίων ενός λεκτικού που αποδίδει ασαφή γνώση σε μία ιδιότητα. Αν η τιμή number ισούται με δύο εμφανίζεται γραφικό διάγραμμα συμπεριλαμβανομένου όλων των λεκτικών που αναφέρονται σε αυτήν την ιδιότητα και αποδίδουν ασαφή γνώση.

- public String getRelation(String value)

Πρόκειται για μία μέθοδο που καλείται από την μέθοδο createChart και επιστρέφει την ιδιότητα για την οποία επιλέχτηκε η εμφάνιση του γραφικού διαγράμματος. Η μεταβλητή value εκτός από την ιδιότητα εμπεριέχει και το λεκτικό στο οποία αναφέρεται για αυτό ακριβώς γίνεται ένας διαχωρισμός της ιδιότητας και του λεκτικού.

- public String getObject(String value)

Πρόκειται για μία μέθοδο που καλείται από την μέθοδο createChart και αντί να επιστρέφει την ιδιότητα όπως πραγματοποιεί η μέθοδος getRelation(), επιστρέφει την τιμή της ιδιότητας που είναι λεκτικό και που προκύπτει από την μεταβλητή value.

- public void resetComponents()

Επειδή μπορεί ένας χρήστης να έχει προβεί ήδη στην ασαφοποίηση μιας crisp οντολογίας και να επιθυμεί να προβεί στην ασαφοποίση μιας άλλης οντολογίας, η συγκεκριμένη μέθοδος επαναφέρει τα components στην αρχική τους κατάσταση. Καλείται από την μέθοδο onParseBtextCkick().

- public static void setVector(String uri, String method)

Η συγκεκριμένη μέθοδος καλείται από το Servlet Step1Servlet.java και ενημερώνει το Vector selections σχετικά με τις ασαφή στοιχεία που επιλέγει ο χρήστης. Επίσης αρχικοποιείται το αντικείμενο step1AdminComponents και καλείται η μέθοδος enableComponents του συγκεκριμένου αντικειμένου. Το uri αντιστοιχεί στο στοιχείο που επέλεξε ο χρήστης στον RDF γράφο και η τιμή της μεταβλητής method στο αν θα πρέπει να αφαιρεθεί ή να προστεθεί αυτή η επιλογή στο Vector selections.

- public static void setSelecteditems(String selection, String Step)

Η συγκεκριμένη μέθοδος μπορεί να κληθεί από τα Servlet Step2aServlet.java, Step2bServlet.java και Step2cServlet. Η τιμή της μεταβλητής Step καθορίζει από πιο Servlet κλήθηκε η συγκεκριμένη μέθοδος και στην συνέχεια αρχικοποιεί ένα από τα αντικείμενα

step2aAdminComponents, step2bAdminComponents και step2cAdminComponents ανάλογά με την τιμή της μεταβλητής Step. Στην συνέχεια καλείται ανάλογα με το αντικείμενο το οποίο έχει αρχικοποιηθεί μία από τις μεθόδους των αντικειμένων runStep2a(),runStep2b ή runStep2c.

- public static void onClearBtextCkick ()

Η συγκεκριμένη μέθοδος καλείται με το πάτημα του κουμπιού που αντιστοιχεί στο component clearBText. Το αποτέλεσμα αυτής ενεργείας είναι ο καθαρισμός του πεδίου που εισάγεται ο rdf κώδικας και αντιστοιχεί στο component rdfText.

- public static void enableButtons ()

Πρόκειται για την μέθοδο που ενεργοποιεί τα components exportButton και nextButton που περικλείονται στο component step1panel (τύπου tabpanel). Η μέθοδος αυτή καλείται από την μέθοδο onParseBtextCkick().

- public static void onParseBtextCkick ()

Η συγκεκριμένη μέθοδος εκτελείται με το πάτημα του κουμπιού που αντιστοιχεί στο component parseBText. Αρχικά γίνεται έλεγχος αν έχει οριστεί έγκυρος XML κώδικας στο component rdfText. Σε διαφορετική περίπτωση εμφανίζεται μήνυμα λάθους. Στην συνέχεια καλείται η μέθοδος resetComponents και getData(), οπού η εκτέλεση της μεθόδου getData() προκαλεί την αρχικοποίηση της μεταβλητής xmlData. Έπειτα καλείται η μέθοδος createDotFile() της κλάσης OntoVisualize, μέσω της οποίας παράγεται το αρχείο XHTML, το οποίο ορίζεται ως πηγή στο component step1Iframe τύπου Iframe. Το συγκεκριμένο XHTML αρχείο ενσωματώνει τον RDF γράφο σε μορφή SVG και τις απαραίτητες συναρτήσεις JavaScript που επιτρέπουν την διαδραστικότητα με τον χρήστη. Τέλος καλούνται οι μέθοδοι enableButton(), editSelection(vect_2), initializeListboxesStep2b() ή initializeListboxesStep2c, μόνο εφόσον ο κώδικας XML που εισήγαγε ο χρήστης περιέχει πληροφορίες για επιλεγμένα ασαφή στοιχεία ή έχει προβεί ήδη στην περιγραφή ασαφών σχέσεων, ασαφών εννοιών ή ασαφών λεκτικών στοιχείων που αποτελούν τιμή σε μία ιδιότητα αντίστοιχα.

- public HashMap getData(String rdfText)

Η συγκεκριμένη μέθοδος αρχικοποιεί την μεταβλητή xmlData. Δοθέντος XML κώδικα που αντιστοιχεί στην τιμή της μεταβλητής rdfText, γίνεται έλεγχος αν περιέχονται πληροφορίες για επιλεγμένα ασαφή στοιχεία, αν έχει προβεί ο χρήστης σε προγενέστερη στιγμή, στην περιγραφή ασαφών σχέσεων, ασαφών εννοιών ή ασαφών λεκτικών. Στην συνέχεια ταξινομεί τις πληροφορίες αυτές στην μεταβλητή xmlData ορίζοντας συγκεκριμένα «κλειδία» ανάλογα με το που αναφέρονται. Το κλειδί main αντιστοιχεί στον κώδικα της crisp οντολογίας, το κλειδί step1 στις πληροφορίες επιλεγμένων ασαφών στοιχείων, το κλειδί step2a στις πληροφορίες ασαφών σχέσεων, το κλειδί step2b στις πληροφορίες ασαφών εννοιών και τέλος το κλειδί step2c στις πληροφορίες ασαφών λεκτικών.

- public void exportStep1()

Η συγκεκριμένη μέθοδος καλείται όταν ο χρήστης πατήσει το κουμπί που αντιστοιχεί στο component exportButton. Στόχος της συγκεκριμένης μεθόδου είναι η εξαγωγή όλων των επιλογών και των στοιχείων που έχει πραγματοποιήσει και ορίσει ο χρήστης αντίστοιχα, σε μορφή XML. Ορίζει την τιμή step1 στην μεταβλητή step για να δηλώσει ότι βρισκόμαστε στο step1panel. Στην συνέχεια γίνεται έλεγχος για το πια tab είναι εμφανίσιμα ώστε να τρέξουν οι κατάλληλοι μέθοδοι (exportStep2aTmp, exportStep2bTmp, exportStep2cTmp) για την συλλογή στοιχείων και από τα αλλά Tabpanel. Έπειτα συλλέγονται οι πληροφορίες από το επιλεγμένο tab. Όλες οι πληροφορίες που συλλέγονται, διαμορφώνονται να έχουν XML δομή και στην συνέχεια αποθηκεύονται μαζί με τον κώδικα της crisp οντολογίας στην μεταβλητή stepXML και τέλος καλείται το servlet XmlServlet.java.

- public void exportStep1Tmp()

Στόχος της συγκεκριμένης μεθόδου είναι η συλλογή και η αποθήκευση των πληροφοριών που αφορούν τα επιλεγμένα ασαφή στοιχεία και που πραγματοποιήθηκε η επιλογή τους στο step1panel με μορφή XML, στην μεταβλητή stepXML.

- public void exportStep2a()

Η συγκεκριμένη μέθοδος καλείται όταν ο χρήστης πατήσει το κουμπί με τίτλο «Export to XML» που βρίσκεται στο step2apanel. Στόχος της συγκεκριμένης μεθόδου είναι η εξαγωγή όλων των επιλογών και των στοιχείων που έχει πραγματοποιήσει και ορίσει ο χρήστης αντίστοιχα, σε μορφή XML. Ορίζει την τιμή step2a στην μεταβλητή step για να δηλώσει ότι βρισκόμαστε στο step2apanel. Στην συνέχεια γίνεται έλεγχος για το πια tab είναι εμφανίσιμα ώστε να τρέξουν οι κατάλληλοι μέθοδοι (exportStep1Tmp, exportStep2bTmp, exportStep2cTmp) για την συλλογή στοιχείων και από τα αλλά Tabpanel. Έπειτα συλλέγονται οι πληροφορίες από το επιλεγμένο tab. Όλες οι πληροφορίες που συλλέγονται, διαμορφώνονται να έχουν XML δομή και στην συνέχεια αποθηκεύονται μαζί με τον κώδικα της crisp οντολογίας στην μεταβλητή stepXML και τέλος καλείται το servlet XmlServlet.java.

- public void exportStep2aTmp()

Στόχος της συγκεκριμένης μεθόδου είναι η συλλογή και η αποθήκευση των πληροφοριών που αφορούν τις πληροφορίες που έχουν οριστεί και συντελούν στην ασαφοποίηση των ασαφών σχέσεων στο step2apanel με μορφή XML, στην μεταβλητή stepXML.

- public void exportStep2b()

Η συγκεκριμένη μέθοδος καλείται όταν ο χρήστης πατήσει το κουμπί με τίτλο «Export to XML» που βρίσκεται στο step2bpanel. Στόχος της συγκεκριμένης μεθόδου είναι η εξαγωγή όλων των επιλογών και των στοιχείων που έχει πραγματοποιήσει και ορίσει ο χρήστης αντίστοιχα, σε μορφή XML. Ορίζει την τιμή step2b στην μεταβλητή step για να δηλώσει ότι βρισκόμαστε στο step2bpanel. Στην συνέχεια γίνεται έλεγχος για το πια tab είναι εμφανίσιμα ώστε να τρέξουν οι κατάλληλοι μέθοδοι (exportStep1Tmp,

exportStep2aTmp, exportStep2cTmp) για την συλλογή στοιχείων και από τα αλλά Tabpanel. Έπειτα συλλέγονται οι πληροφορίες από το επιλεγμένο tab. Όλες οι πληροφορίες που συλλέγονται, διαμορφώνονται να έχουν XML δομή και στην συνέχεια αποθηκεύονται μαζί με τον κώδικα της crisp οντολογίας στην μεταβλητή stepXML και τέλος καλείται το servlet XmlServlet.java.

- public void exportStep2bTmp()

Στόχος της συγκεκριμένης μεθόδου είναι η συλλογή και η αποθήκευση των πληροφοριών που αφορούν τις πληροφορίες που έχουν οριστεί και συντελούν στην ασαφοποίηση των ασαφών εννοιών στο step2bpanel με μορφή XML, στην μεταβλητή stepXML.

- public void exportStep2c()

Η συγκεκριμένη μέθοδος καλείται όταν ο χρήστης πατήσει το κουμπί με τίτλο «Export to XML» που βρίσκεται στο step2cpanel. Στόχος της συγκεκριμένης μεθόδου είναι η εξαγωγή όλων των επιλογών και των στοιχείων που έχει πραγματοποιήσει και ορίσει ο χρήστης αντίστοιχα, σε μορφή XML. Ορίζει την τιμή step2c στην μεταβλητή step για να δηλώσει ότι βρισκόμαστε στο step2bpanel. Στην συνέχεια γίνεται έλεγχος για το πια tab είναι εμφανίσιμα ώστε να τρέξουν οι κατάλληλοι μέθοδοι (exportStep1Tmp, exportStep2aTmp, exportStep2bTmp) για την συλλογή στοιχείων και από τα αλλά Tabpanel. Έπειτα συλλέγονται οι πληροφορίες από το επιλεγμένο tab. Όλες οι πληροφορίες που συλλέγονται, διαμορφώνονται να έχουν XML δομή και στην συνέχεια αποθηκεύονται μαζί με τον κώδικα της crisp οντολογίας στην μεταβλητή stepXML και τέλος καλείται το servlet XmlServlet.java.

- public void exportStep2cTmp()

Στόχος της συγκεκριμένης μεθόδου είναι η συλλογή και η αποθήκευση των πληροφοριών που αφορούν τις πληροφορίες που έχουν οριστεί και συντελούν στην ασαφοποίηση των ασαφών λεκτικών που αποτελούν τιμή σε μία ιδιότητα στο step2bpanel με μορφή XML, στην μεταβλητή stepXML.

- public void initializeVectors ()

Η συγκεκριμένη μέθοδός εισάγει τις ασαφείς έννοιες που έχει επιλέξει ο χρήστης στο Vector vect_1, τις ασαφείς σχέσεις που έχει επιλέξει ο χρήστης στο vect_2 και τα ασαφή λεκτικά που αποτελούν τιμές σε μία ιδιότητα στο vect_3.

- public void initializeVectors ()

Η συγκεκριμένη μέθοδός εισάγει τις ασαφείς έννοιες που έχει επιλέξει ο χρήστης στο Vector vect_1, τις ασαφείς σχέσεις που έχει επιλέξει ο χρήστης στο vect_2 και τα ασαφή λεκτικά που αποτελούν τιμές σε μία ιδιότητα στο vect_3.

- public void goTostep2()

Η συγκεκριμένη μέθοδός καλείται από το component nextButton που περικλείεται μέσα στο component step1panel. Αρχικά γίνεται έλεγχος αν το μέγεθός του Vector selections είναι μεγαλύτερο από το μηδέν (το Vector selections περιέχει τις επιλογές του χρήστη για ασαφή στοιχεία) και εφόσον είναι στην συνέχεια καλείται η μέθοδος initializeVectors(). Έπειτα αρχικοποιούνται τα components τύπου Listbox relationListbox, relationStatement, conceptInstances, conceptListbox και fuzzyDatatypes. Τέλος γίνεται έλεγχος αν ο χρήστης έχει επιλέξει ασαφείς σχέσεις (vect_2.size()>0), σε αυτήν την περίπτωση εμφανίζει το tabstep2a και καλεί την μέθοδο editSelection(). Διαφορετικά κάνει έλεγχο αν ο χρήστης έχει επιλέξει ασαφείς έννοιες (vect_1.size()>0) και εμφανίζει το tabstep2b και καλεί την μέθοδο initializeListboxesStep2b() ή αν δεν ισχύει καμία από τις παραπάνω περιπτώσεις εμφανίζει το tabstep2c και καλεί την μέθοδο initializeListboxesStep2c().

- public void goTostep2b()

Η συγκεκριμένη μέθοδος καλείται όταν ο χρήστης πατήσει το κουμπί με τίτλο «Next Step» που βρίσκεται στο step2apanel. Στόχος της συγκεκριμένης μεθόδου, είναι αρχικά να πραγματοποιηθεί έλεγχος εγκυρότητας των πεδίων των εγγραφών που υπάρχουν στο συγκεκριμένο πάνελ, κάνοντας χρήση της μεθόδου checkStep2a(). Στην συνέχεια ανάλογα με τα ασαφή στοιχεία που είχε επιλέξει στο βήμα «step1», μεταφέρεται στο step2bpanel όπου μπορεί να προβεί στην ασαφοποίηση των εννοιών που είχε επιλέξει ή μεταφέρεται στο step2cpanel όπου μπορεί να προβεί στην ασαφοποίηση των λεκτικών που αποτελούν τιμές στις ιδιότητες που είχε επιλέξει.

- public void goTostep2b()

Η συγκεκριμένη μέθοδος καλείται όταν ο χρήστης πατήσει το κουμπί με τίτλο «Next Step» που βρίσκεται στο step2bpanel. Στόχος της συγκεκριμένης μεθόδου, είναι αρχικά να πραγματοποιηθεί έλεγχος εγκυρότητας των πεδίων των εγγραφών που υπάρχουν στο συγκεκριμένο πάνελ, κάνοντας χρήση της μεθόδου checkStep2b(). Στην συνέχεια εφόσον ο χρήστης στο βήμα «step1» έχει επιλέξει ιδιότητες οπού οι τιμές του αποδίδουν ασαφή γνώση, μεταφέρεται στο step2cpanel όπου μπορεί να προβεί στην ασαφοποίηση των λεκτικών που αποτελούν τιμές στις ιδιότητες που επέλεξε.

- public void resetFieldsStep2a()

Η συγκεκριμένη μέθοδός επαναφέρει στην αρχική τους κατάσταση τα components που είναι τύπου Textbox και Doublebox και που ανήκουν στις εγγραφές των Listboxes relationListbox και relationStatement αντίστοιχα. Η μέθοδος resetFieldsStep2a() καλείται από την μέθοδο exportStep2a().

- public void resetFieldsStep2b()

Η συγκεκριμένη μέθοδός επαναφέρει στην αρχική τους κατάσταση τα components που είναι τύπου Textbox και Doublebox και που ανήκουν στις εγγραφές των Listboxes

conceptListbox και conceptInstances αντίστοιχα. Η μέθοδος resetFieldsStep2b() καλείται από την μέθοδο exportStep2b().

- public void resetstep2cListitem (Listitem listitem)

Η συγκεκριμένη μέθοδός επαναφέρει στην αρχική τους κατάσταση τα components που είναι τύπου Doublebox και που ανήκουν στο αντικείμενο της listitem του Listbox fuzzyDatatypes. Η μέθοδος resetstep2cListitem (Listitem listitem) καλείται από την μέθοδο createChart(int number).

- public void checkStep2a()

Πρόκειται για την μέθοδό που εξετάσει αν τιμές που έχουν εκχωρηθεί στα components τύπου Textbox και Doublebox και που ανήκουν στις εγγραφές των Listboxes relationListbox και relationStatement αντίστοιχα, είναι έγκυρες. Σε περίπτωση λάθους επιστρέφει μια μεταβλητή Boolean με τιμή true. Η μέθοδος checkStep2a() καλείται από την μέθοδο exportStep2a().

- public void checkStep2b()

Πρόκειται για την μέθοδό που εξετάσει αν τιμές που έχουν εκχωρηθεί στα components τύπου Textbox και Doublebox και που ανήκουν στις εγγραφές των Listboxes conceptListbox και conceptInstances αντίστοιχα, είναι έγκυρες. Σε περίπτωση λάθους επιστρέφει μια μεταβλητή Boolean με τιμή true. Η μέθοδος checkStep2b() καλείται από την μέθοδο exportStep2b().

- public void checkStep2c()

Πρόκειται για την μέθοδό που εξετάσει αν τιμές που έχουν εκχωρηθεί στα components τύπου Doublebox και που ανήκουν στις εγγραφές του Listbox fuzzyDatatypes, είναι έγκυρες. Σε περίπτωση λάθους επιστρέφει μια μεταβλητή Boolean με τιμή true. Η μέθοδος checkStep2c() καλείται από την μέθοδο exportStep2c().

- public void setStep2aData()

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο editSelection(). Αρχικά ελέγχει αν υπάρχουν πληροφορίες που να έχει ορίσει ο χρήστης σε προγενέστερη περίοδο και που αφορούν ασαφείς σχέσεις. Αυτό επιτυγχάνεται με το να ελέγχει αν υπάρχει εγγραφή στο Hashmap xmlData με κλειδί «step2a». Εφόσον υπάρχουν πληροφορίες δημιουργεί έγγραφες στα Listboxes relationListbox και relationStatement με τις πληροφορίες αυτές.

- public void setStep2bData()

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο initializeListboxesStep2b(). Αρχικά ελέγχει αν υπάρχουν πληροφορίες που να έχει ορίσει ο χρήστης σε προγενέστερη περίοδο και που αφορούν ασαφείς έννοιες. Αυτό επιτυγχάνεται με το να ελέγχει αν υπάρχει εγγραφή στο Hashmap xmlData με κλειδί «step2b». Εφόσον υπάρχουν πληροφορίες δημιουργεί έγγραφες στα Listboxes conceptListbox και conceptInstances με τις πληροφορίες αυτές.

- public void setStep2cData()

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο initializeListboxesStep2c(). Αρχικά ελέγχει αν υπάρχουν πληροφορίες που να έχει ορίσει ο χρήστης σε προγενέστερη περίοδο και που αφορούν ασαφή λεκτικά που αποτελούν τιμή σε μία ιδιότητα. Αυτό επιτυγχάνεται με το να ελέγχει αν υπάρχει εγγραφή στο Hashmap xmlData με κλειδί «step2c». Εφόσον υπάρχουν πληροφορίες δημιουργεί έγγραφες στο Listbox fuzzyDatatypes με τις πληροφορίες αυτές.

- public void setStep2cData()

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο initializeListboxesStep2c(). Αρχικά ελέγχει αν υπάρχουν πληροφορίες που να έχει ορίσει ο χρήστης σε προγενέστερη περίοδο και που αφορούν ασαφή λεκτικά που αποτελούν τιμή σε μία ιδιότητα. Αυτό επιτυγχάνεται με το να ελέγχει αν υπάρχει εγγραφή στο Hashmap xmlData με κλειδί «step2c». Εφόσον υπάρχουν πληροφορίες δημιουργεί έγγραφες στο Listbox fuzzyDatatypes με τις πληροφορίες αυτές.

- public String getCharacterDataFromElement(Element e)

Η συγκεκριμένη μέθοδος μπορεί να κληθεί από τις μεθόδους setStep2aData(), setStep2bData() και setStep2cData(). Δοθέντος ενός XML element ελέγχει αν το συγκεκριμένο elemenet είναι στιγμιότυπο του αντικειμένου CharacterData και εφόσον είναι επιστρέφει την τιμή του.

- public void editSelection(Vector selections)

Πρόκειται για την μέθοδο η οποία αρχικοποιεί τα Listboxes relationListbox και relationStatement. Συγκεκριμένα στο relationListbox δίνεται η ευχέρεια στον χρήστη να ορίσει το «Vague Nature» και «Degree Interpetation» για κάθε ασαφή σχέση που έχει επιλέξει, ενώ στο Listbox relationStatement ο χρήστης ορίζει τους βαθμούς ασάφειας για την κάθε επιλεγμένη ασαφή σχέση σε συνδυασμό με τα στιγμιότυπα εννοιών που συνδέονται με αυτήν την σχέση. Ο αριθμός των εγγραφών και κατά συνέπεια των αντικειμένων Listitem που έχουν τα παραπάνω Listboxes, είναι άρρηκτα συνδεμένος με το μέγεθος του Vector selections. Το Vector selection περιέχει τις επιλεγμένες ασαφείς σχέσεις. Σε κάθε εγγραφή του Listbox relationListbox δημιουργούνται αντικείμενα τύπου Textbox που αντιστοιχούν στην τιμή «Vague Nature» και «Degree Interpetation» για την επιλεγμένη σχέση, ενώ σε κάθε εγγραφή του Listbox relationStatement δημιουργείται αντικείμενο τύπου Doublebox που αντιστοιχεί στον βαθμό ασάφειας της επιλεγμένης σχέσης. Τέλος καλείται η μέθοδος updateSvgFile() της κλάσης UpdateSvgFileStep2a, μέσω της οποίας παράγεται το αρχείο XHTML, το οποίο ορίζεται ως πηγή στο component step2aIframe τύπου Iframe. Το συγκεκριμένο XHTML αρχείο ενσωματώνει τον RDF γράφο σε μορφή SVG, το οποίο έχει επιλεγμένες τις ασαφείς σχέσεις που είχε επιλέξει ο χρήστης σε προηγούμενο βήμα και για τις οποίες επιθυμεί να προβεί στην ασαφοποίηση τους. Επίσης το XHTML αρχείο ενσωματώνει τις απαραίτητες συναρτήσεις JavaScript που επιτρέπουν την διαδραστικότητα με τον χρήστη. Η μέθοδος ολοκληρώνεται με την εμφάνιση του tab Step2a.

- public void initializeListboxesStep2b()

Πρόκειται για την μέθοδο η οποία αρχικοποιεί τα Listboxes conceptListbox και conceptInstances. Συγκεκριμένα στο conceptListbox δίνεται η ευχέρεια στον χρήστη να ορίσει το «Vague Nature» και «Degree Interpetation» για κάθε ασαφή έννοια που έχει επιλέξει, ενώ στο Listbox conceptInstances ο χρήστης ορίζει τους βαθμούς ασάφειας για την κάθε επιλεγμένη ασαφή έννοια. Ο αριθμός των εγγραφών και κατά συνέπεια των αντικειμένων Listitem που έχουν τα παραπάνω Listboxes, είναι άρρηκτα συνδεμένος με το μέγεθος του Vector vect_1. Το Vector vect_1 περιέχει τις επιλεγμένες ασαφείς έννοιες και έχει αρχικοποιηθεί με την κλήση της μεθόδου initializeVectors(). Σε κάθε εγγραφή του Listbox conceptListbox δημιουργούνται τα αντικείμενα τύπου Textbox που αντιστοιχούν στην τιμή «Vague Nature» και «Degree Interpetation» για την επιλεγμένη έννοια, ενώ σε κάθε εγγραφή του Listbox conceptInstances δημιουργείται αντικείμενο τύπου Doublebox που αντιστοιχεί στον βαθμό ασάφειας της επιλεγμένης έννοιας. Τέλος καλείται η μέθοδος updateSvgFile() της κλάσης UpdateSvgFileStep2b, μέσω της οποίας παράγεται το αρχείο XHTML, το οποίο ορίζεται ως πηγή στο component step2bIframe τύπου Iframe. Το συγκεκριμένο XHTML αρχείο ενσωματώνει τον RDF γράφο σε μορφή SVG, το οποίο έχει επιλεγμένες τις ασαφείς έννοιες που είχε επιλέξει ο χρήστης σε προηγούμενο βήμα και για τις οποίες επιθυμεί να προβεί στην ασαφοποίηση τους. Επίσης το XHTML αρχείο ενσωματώνει τις απαραίτητες συναρτήσεις JavaScript που επιτρέπουν την διαδραστικότητα με τον χρήστη. Η μέθοδος ολοκληρώνεται με την εμφάνιση του tab Step2b.

- public void initializeListboxesStep2C()

Πρόκειται για την μέθοδο η οποία αρχικοποιεί το Listbox fuzzyDatatypes. Συγκεκριμένα στο fuzzyDatatypes δίνεται η ευχέρεια στον προσδιορίσει τα όρια (πεδίο τιμών) ενός λεκτικού που αναπαριστά ασαφή γνώση ως τιμή μιας ιδιότητας, συμπληρώνοντας τα πεδία που απαιτούνται. Ο αριθμός των εγγραφών και κατά συνέπεια των αντικειμένων Listitem που έχει το παραπάνω Listbox, είναι άρρηκτα συνδεμένος με το μέγεθος του Vector vect_3. Το Vector vect_3 περιέχει τις επιλεγμένες ασαφείς ιδιότητες σε συνδυασμό με τα λεκτικά που αποτελούν τιμές τους και έχει αρχικοποιηθεί με την κλήση της μεθόδου initializeVectors(). Σε κάθε εγγραφή του Listbox fuzzyDatatypes δημιουργούνται αντικείμενα τύπου Doublebox που αντιστοιχούν στο πεδίο τιμών του λεκτικού που παρουσιάζει ασάφεια ως τιμή της επιλεγμένης ιδιότητας. Σε κάθε αντικείμενο Doublebox εκχωρείται ένας διαχείρισης γεγονότων (Event Listener), έτσι ώστε ο χρήστης μόλις εστιάσει σε αντικείμενο Doublebox, να του απεικονίζει το κατάλληλο γράφημα για το πώς θα πρέπει να συμπληρώσει το πεδίο, το οποίο αντιστοιχεί στο αντικείμενο Doublebox. Τέλος καλείται η μέθοδος updateSvgFile() της κλάσης UpdateSvgFileStep2c, μέσω της οποίας παράγεται το αρχείο XHTML, το οποίο ορίζεται ως πηγή στο component step2cIframe τύπου Iframe. Το συγκεκριμένο XHTML αρχείο ενσωματώνει τον RDF γράφο σε μορφή SVG, το οποίο έχει επιλεγμένες τις ιδιότητες και τα ασαφή λεκτικά που αποτελούν τιμές τους, που είχε επιλέξει ο χρήστης σε προηγούμενο βήμα και για τα οποία επιθυμεί να προβεί στην ασαφοποίηση τους. Επίσης το XHTML αρχείο ενσωματώνει τις απαραίτητες συναρτήσεις JavaScript που επιτρέπουν την διαδραστικότητα με τον χρήστη. Η μέθοδος ολοκληρώνεται με την εμφάνιση του tab Step2c.

7.3.2 *STEP1SERVLET.JAVA*

Δεδομένα Εισόδου:

- **uri**. Αντιστοιχεί στο επιλεγμένο στοιχείο.
- **method**. Η τιμή της μεταβλητής method μας δείχνει αν θα πρέπει να προσθέσουμε ένα επιλεγμένο στοιχείο στα ήδη επιλεγμένα στοιχεία ή αν θα πρέπει να αφαιρέσουμε ένα στοιχείο από τα επιλεγμένα στοιχεία. Οι τιμές που μπορεί να εκχωρηθούν στην μεταβλητή method είναι οι εξής:
  - **add1**. Δηλώνει ότι το στοιχείο που έχει επιλεχτεί είναι ασαφή έννοια και ότι θα πρέπει να προστεθεί στα επιλεγμένα στοιχεία.
  - **delete1**. Δηλώνει ότι το στοιχείο που έχει επιλεχτεί είναι ασαφή έννοια και ότι θα πρέπει να αφαιρεθεί από τα επιλεγμένα στοιχεία.
  - **add2**. Δηλώνει ότι το στοιχείο που έχει επιλεχτεί είναι ασαφή σχέση και ότι θα πρέπει να προστεθεί στα επιλεγμένα στοιχεία.
  - **delete2**. Δηλώνει ότι το στοιχείο που έχει επιλεχτεί είναι ασαφή σχέση και ότι θα πρέπει να αφαιρεθεί από τα επιλεγμένα στοιχεία.
  - **add3**. Δηλώνει ότι το στοιχείο που έχει επιλεχτεί είναι ιδιότητα σε συνδυασμό με το ασαφή λεκτικό το οποίο αποτελεί τιμή της ιδιότητας και ότι θα πρέπει να προστεθούν στα επιλεγμένα στοιχεία.
  - **delete3**. Δηλώνει ότι το στοιχείο που έχει επιλεχτεί είναι ιδιότητα σε συνδυασμό με το ασαφή λεκτικό το οποίο αποτελεί τιμή της ιδιότητας και ότι θα πρέπει να αφαιρεθούν από τα επιλεγμένα στοιχεία.

Περιγραφή Λειτουργίας:

Το Step1Servlet.java καλείται μόλις ο χρήστης πραγματοποιήσει κάποια επιλογή ενός στοιχείου που αποδίδει ασαφή γνώση στον RDF γράφο που απεικονίζεται, όταν ο χρήστης βρίσκεται στο βήμα «step1». Μέσα από το συγκεκριμένο servlet καλείται κάθε φόρα η μέθοδος setVector(String uri, String method) της κλάσης ConstMainWin.java.

Κλήσεις:

```
import zul.ConstMainWin;
import java.io.*;
import java.util.Vector;
import javax.servlet.*;
import javax.servlet.http.*;
```

7.3.3 *STEP2ASEVLET.JAVA*

Δεδομένα Εισόδου:

- **uri**. Αντιστοιχεί στο επιλεγμένο στοιχείο.

Περιγραφή Λειτουργίας:

Το Step2aServlet.java καλείται μόλις ο χρήστης πραγματοποιήσει κάποια επιλογή μίας σχέσης που αποδίδει ασαφή γνώση στον RDF γράφο που απεικονίζεται, όταν ο χρήστης βρίσκεται στο βήμα «step2a». Μέσα από το συγκεκριμένο servlet καλείται κάθε φόρα η μέθοδος setSelecteditems(String selection, String Step) της κλάσης ConstMainWin.java, οπού η τιμή της μεταβλητής selection αντιστοιχεί στην τιμή της μεταβλητής uri και η τιμή της μεταβλητής Step ισούται με step2a. Αυτό που επιθυμούμε να επιτύχουμε είναι μόλις ο χρήστης επιλέξει ένα στοιχείο, να εστιάσει στις έγγραφες που υπάρχουν στα Listboxes relationListbox και relationStatement της σελίδας index.zul και που αναφέρονται στο επιλεγμένο στοιχείο.

Κλήσεις:

```
import zul.ConstMainWin;
import java.io.*;
import java.util.Vector;
import javax.servlet.*;
import javax.servlet.http.*;
```

7.3.4 *STEP2BSEVLET.JAVA*

Δεδομένα Εισόδου:

- **uri**. Αντιστοιχεί στο επιλεγμένο στοιχείο.

Περιγραφή Λειτουργίας:

Το Step2bServlet.java καλείται μόλις ο χρήστης πραγματοποιήσει κάποια επιλογή μίας έννοιας που αποδίδει ασαφή γνώση στον RDF γράφο που απεικονίζεται, όταν ο χρήστης βρίσκεται στο βήμα «step2b». Μέσα από το συγκεκριμένο servlet καλείται κάθε φόρα η μέθοδος setSelecteditems(String selection, String Step) της κλάσης ConstMainWin.java, οπού η τιμή της μεταβλητής selection αντιστοιχεί στην τιμή της μεταβλητής uri και η τιμή της μεταβλητής Step ισούται με step2b. Αυτό που επιθυμούμε να επιτύχουμε είναι μόλις ο χρήστης επιλέξει ένα στοιχείο, να εστιάσει στις έγγραφες που υπάρχουν στα Listboxes conceptListbox και conceptInstances της σελίδας index.zul και που αναφέρονται στο επιλεγμένο στοιχείο.

Κλήσεις:

```
import zul.ConstMainWin;
import java.io.*;
import java.util.Vector;
import javax.servlet.*;
import javax.servlet.http.*;
```

7.3.5 *STEP2CSEVLET.JAVA*

Δεδομένα Εισόδου:

- **uri**. Αντιστοιχεί στο επιλεγμένο στοιχείο.

Περιγραφή Λειτουργίας:

Το Step2cServlet.java καλείται μόλις ο χρήστης πραγματοποιήσει μία επιλογή μίας ιδιότητας πάνω στον RDF γράφο που απεικονίζεται, όπου σε συνδυασμό με την τιμή της που είναι λεκτικό, αποδίδουν ασαφή γνώση. Ο χρήστης βρίσκεται στο βήμα «step2c». Μέσα από το συγκεκριμένο servlet καλείται κάθε φόρα η μέθοδος setSelecteditems(String selection, String Step) της κλάσης ConstMainWin.java, οπού η τιμή της μεταβλητής selection αντιστοιχεί στην τιμή της μεταβλητής uri και η τιμή της μεταβλητής Step ισούται με step2c. Αυτό που επιθυμούμε να επιτύχουμε είναι μόλις ο χρήστης επιλέξει ένα στοιχείο, να εστιάσει στην εγγραφή που υπάρχει στο Listbox fuzzyDatatypes της σελίδας index.zul και που αναφέρονται στο επιλεγμένο στοιχείο.

Κλήσεις:

```
import zul.ConstMainWin;
import java.io.*;
import java.util.Vector;
import javax.servlet.*;
import javax.servlet.http.*;
```

7.3.6 *XMLSERVLET.JAVA*

Δεδομένα Εισόδου:

- **ConstMainWin.stepXML**. Η αντίστοιχη μεταβλητή της κλάσης ConstMainWin.java.

Περιγραφή Λειτουργίας:

Το XmlServlet.java είναι υπεύθυνο για την εξαγωγή όλων των επιλογών που έχει πραγματοποιήσει ο χρήστης σε μορφή XML, οπού θα μπορεί η συγκεκριμένη μορφή να είναι επεξεργάσιμη από την πλατφόρμα ασαφοποίησης και σε μεταγενέστερες περιόδους.

Κλήσεις:

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
```

7.3.7 *STEP1ADMINCOMPONENTS.JAVA*

Περιγραφή Λειτουργίας:

Η κλάση Step1AdminComponents.java είναι υπεύθυνη για την ενεργοποίηση των κουμπιών exportButton και nextButton, της σελίδας index.zul. Μόλις ο χρήστης επιλέξει μια επιλογή ασαφούς στοιχείου στο βήμα step1, τα κουμπιά ενεργοποιούνται. Σε κάθε επιλογή που πραγματοποιεί ο χρήστης γίνεται έλεγχος για την ύπαρξη εμφανίσιμων tab και σε περίπτωση που υπάρχουν τα απενεργοποιεί.

Κλήσεις:

```
import org.zkoss.zk.ui.Desktop;
import org.zkoss.zk.ui.Executions;
import org.zkoss.zul.*;
```

Μεταβλητές Κλάσης:

```
private static Desktop desktop;  -> Αντικείμενο τύπου Desktop
private static Button nextButton;-> Αντικείμενο τύπου Button
private static Button exportButton;-> Αντικείμενο τύπου Button
private static Tab tabstep2a; -> Αντικείμενο τύπου Tab
private static Tab tabstep2b; -> Αντικείμενο τύπου Tab
private static Tab tabstep2c; -> Αντικείμενο τύπου Tab
```

Μέθοδοι:

- public Step1AdminComponents(Button nextButton, Button exportButton,Tab tabstep2a,Tab tabstep2b,Tab tabstep2c)

Πρόκειται για τον Constructor της συγκεκριμένης κλάσης. Στην συγκεκριμένη μέθοδο αρχικοποιείται και το αντικείμενο desktop. Για να μπορέσουμε να διαχειριστούμε

τα αντικείμενα που έχουμε ορίσει στον Constructor της συγκεκριμένης κλάσης θα πρέπει να ορίσουμε την τιμή true στην μέθοδο enableServerPush(true) του αντικείμενου desktop.

- public void enableComponents(int size,boolean disableTabs)

Αν η τιμή της μεταβλητής size είναι μεγαλύτερη της τιμής 0, τότε ενεργοποιούνται τα κουμπιά exportButton και nextButton, σε διαφορετική περίπτωση απενεργοποιούνται. Η τιμή size μας δείχνει αν ο χρήστης έχει προβεί στην επιλογή ασαφών στοιχείων. Τέλος αν η τιμή της μεταβλητής disableTabs είναι αληθής, τότε σε αυτήν την περίπτωση και μόνο απενεργοποιούνται τα tabs tabstep2a, tabstep2b και tabstep2c της Index.zul σελίδας.

7.3.8 *STEP2AADMINCOMPONENTS.JAVA*

Περιγραφή Λειτουργίας:

Μέσω της κλάση Step2aAdminComponents.java ολοκληρώνεται η δυνατότητα που παρέχετε σε έναν χρήστη και ενώ βρίσκεται στο βήμα step2a, μόλις επιλέξει μία σχέση (από τον RDF γράφο) να εστιάσει στις έγγραφες που υπάρχουν στα Listboxes relationListbox και relationStatement της σελίδας index.zul, οι οποίες αναφέρονται στην επιλεγμένη σχέση.

Κλήσεις:

```
import org.zkoss.zk.ui.Desktop;
import org.zkoss.zk.ui.Executions;
import org.zkoss.zul.*;
import java.util.StringTokenizer;
```

Μεταβλητές Κλάσης:

```
private static Desktop desktop;  -> Αντικείμενο τύπου Desktop
private static Listbox  relationListbox;-> Αντικείμενο τύπου Listbox
private static Listbox  relationStatement;-> Αντικείμενο τύπου Listbox
```

Μέθοδοι:

- public Step2aAdminComponents(Listbox relationListbox, Listbox relationStatement)

Πρόκειται για τον Constructor της συγκεκριμένης κλάσης. Στην συγκεκριμένη μέθοδο αρχικοποιείται και το αντικείμενο desktop. Για να μπορέσουμε να διαχειριστούμε τα αντικείμενα που έχουμε ορίσει στον Constructor της συγκεκριμένης κλάσης θα πρέπει να ορίσουμε την τιμή true στην μέθοδο enableServerPush(true) του αντικείμενου desktop.

- public void runStep2a(String selection)

Μέσω της συγκεκριμένης μεθόδου επιτυγχάνεται η δυνατότητα που παρέχετε σε έναν χρήστη και ενώ βρίσκεται στο βήμα step2a, μόλις επιλέξει μία σχέση (από τον RDF γράφο) να εστιάσει στις έγγραφες που υπάρχουν στα Listboxes relationListbox και relationStatement της σελίδας index.zul , οι οποίες αναφέρονται στην επιλεγμένη σχέση. Η τιμή της μεταβλητής selection αντιστοιχεί στο επιλεγμένο στοιχείο.

7.3.9 *STEP2BADMINCOMPONENTS.JAVA*

Περιγραφή Λειτουργίας:

Μέσω της κλάση Step2bAdminComponents.java ολοκληρώνεται η δυνατότητα που παρέχετε σε έναν χρήστη και ενώ βρίσκεται στο βήμα step2b, μόλις επιλέξει μία ασαφή έννοια (από τον RDF γράφο) να εστιάσει στις έγγραφες που υπάρχουν στα Listboxes conceptListbox και conceptInstances της σελίδας index.zul, οι οποίες αναφέρονται στην επιλεγμένη έννοια.

Κλήσεις:

```
import org.zkoss.zk.ui.Desktop;
import org.zkoss.zk.ui.Executions;
import org.zkoss.zul.*;
import java.util.StringTokenizer;
```

Μεταβλητές Κλάσης:

```
private static Desktop desktop;  -> Αντικείμενο τύπου Desktop
private static Listbox  conceptListbox;-> Αντικείμενο τύπου Listbox
private static Listbox   conceptInstances;-> Αντικείμενο τύπου Listbox
```

Μέθοδοι:

- public Step2bAdminComponents(Listbox conceptListbox, Listbox conceptInstances)

Πρόκειται για τον Constructor της συγκεκριμένης κλάσης. Στην συγκεκριμένη μέθοδο αρχικοποιείται και το αντικείμενο desktop. Για να μπορέσουμε να διαχειριστούμε τα αντικείμενα που έχουμε ορίσει στον Constructor της συγκεκριμένης κλάσης θα πρέπει να ορίσουμε την τιμή true στην μέθοδο enableServerPush(true) του αντικείμενου desktop.

- public void runStep2b(String selection)

Μέσω της συγκεκριμένης μεθόδου επιτυγχάνεται η δυνατότητα που παρέχετε σε έναν χρήστη και ενώ βρίσκεται στο βήμα step2b, μόλις επιλέξει μία σχέση (από τον RDF

γράφο) να εστιάσει στις έγγραφες που υπάρχουν στα Listboxes conceptListbox και conceptInstances της σελίδας index.zul, οι οποίες αναφέρονται στην επιλεγμένη έννοια. Η τιμή της μεταβλητής selection αντιστοιχεί στο επιλεγμένο στοιχείο.

7.3.10 *STEP2CADMINCOMPONENTS.JAVA*

Περιγραφή Λειτουργίας:

Μέσω της κλάση Step2cAdminComponents.java ολοκληρώνεται η δυνατότητα που παρέχετε σε έναν χρήστη και ενώ βρίσκεται στο βήμα step2c, μόλις επιλέξει μία ασαφή ιδιότητα σε συνδυασμό με το λεκτικό που αποτελεί τιμή της, από τον RDF γράφο, να εστιάσει στην έγγραφη που υπάρχει στο Listbox fuzzyDatatypes της σελίδας index.zul, οι οποίες αναφέρονται στα επιλεγμένα στοιχεία.

Κλήσεις:

```
import org.zkoss.zk.ui.Desktop;
import org.zkoss.zk.ui.Executions;
import org.zkoss.zul.*;
```

Μεταβλητές Κλάσης:

```
private static Desktop desktop;-> Αντικείμενο τύπου Desktop
private static Listbox   fuzzyDatatypes;-> Αντικείμενο τύπου Listbox
```

Μέθοδοι:

- public Step2bAdminComponents(Listbox fuzzyDatatypes)

Πρόκειται για τον Constructor της συγκεκριμένης κλάσης. Στην συγκεκριμένη μέθοδο αρχικοποιείται και το αντικείμενο desktop. Για να μπορέσουμε να διαχειριστούμε τα αντικείμενα που έχουμε ορίσει στον Constructor της συγκεκριμένης κλάσης θα πρέπει να ορίσουμε την τιμή true στην μέθοδο enableServerPush(true) του αντικείμενου desktop.

- public void runStep2c(String selection)

Μέσω της συγκεκριμένης μεθόδου επιτυγχάνεται η δυνατότητα που παρέχετε σε έναν χρήστη και ενώ βρίσκεται στο βήμα step2c μόλις επιλέξει μία ασαφή ιδιότητα σε συνδυασμό με το λεκτικό που αποτελεί τιμή της, από τον RDF γράφο, να εστιάσει στην έγγραφη που υπάρχει στο Listbox fuzzyDatatypes της σελίδας index.zul, οι οποίες αναφέρονται στα επιλεγμένα στοιχεία. Η τιμή της μεταβλητής selection αντιστοιχεί στα επιλεγμένα στοιχεία.

7.3.11 *FUNCTIONS.JAVA*

Περιγραφή Λειτουργίας:

Η συγκεκριμένη Java κλάση είναι μία βοηθητική κλάση, η οποία χρησιμοποιείται από τις κλάσεις του πακέτου update. Δεν έχει υλοποιηθεί καμία μέθοδος για την συγκεκριμένη κλάση, έχουν ορισθεί συγκεκριμένες μεταβλητές τύπου String, που αντιστοιχούν στα XHTML στοιχεία και στις συναρτήσεις JavaScript που μαζί με το παραγόμενο αρχείο SVG κάθε φορά που αντιστοιχεί στον RDF γράφο του κάθε βήματος(step1, ste2a, step2c ή step2c) της πλατφόρμας *wb*GraphFuzzyOnto , συνθέτουν το παραγόμενο αρχείο XHTML, που ορίζεται σε κάθε βήμα.

Μεταβλητές Κλάσης:

```
Μεταβλητές τύπου String που χρησιμοποιούνται από την Κλάση UpdateSvgFileStep1
 public static final String step1_head_con;
 public static final String step1_head;
 public static final String step1_end

Μεταβλητές τύπου String που χρησιμοποιούνται από την Κλάση UpdateSvgFileStep2a
public static final String step2a_head
public static final String ste2a_end

Μεταβλητές τύπου String που χρησιμοποιούνται από την Κλάση UpdateSvgFileStep2b
public static final String step2b_head
public static final String step2b_end

Μεταβλητές τύπου String που χρησιμοποιούνται από την Κλάση UpdateSvgFileStep2c
public static final String step2b_head
public static final String step2b_end
```

7.3.12 *UPDATESVGFILESTEP1.JAVA*

Περιγραφή Λειτουργίας:

Πρόκειται για μία από τις κλάσεις, η οποία επεξεργάζεται κατάλληλα το SVG αρχείο που αναπαριστά τον RDF γράφο, ενσωματώνοντας επίσης και JavaScript συναρτήσεις, έτσι ώστε να επιτρέπεται η διαδραστικότητα με τον χρήστη. Η συγκεκριμένη διαδικασία, ολοκληρώνεται με την δημιουργία ενός νέου αρχείου XHTML το οποίο ορίζεται σαν πηγή στο component step1Frame της index.zul σελίδας. Μέσα από την λειτουργικότητα που παρέχεται από το παραγόμενο αρχείο, επιτυγχάνεται η δυνατότητα επιλογής ασαφών στοιχείων στο βήμα step1.

Κλήσεις:

```
import java.util.*;
import java.io.*;
```

Μέθοδοι:

- public HashMap initializeMap(String file)

Σε έναν RDF γράφο και κατά συνέπεια σε ένα αρχείο SVG, μία ακμή αντιστοιχεί στην αναπαράσταση μίας σχέσης ή μίας ιδιότητας. Κάθε ακμή έχει ένα ειδικό αναγνωριστικό το οποίο αντιστοιχεί στην τιμή edge και ενός μοναδικού αριθμού για αυτήν την ακμή (π.χ. edge14). Στην συγκεκριμένη μέθοδο επιστρέφουμε ένα αντικείμενο τύπου HashMap, ορίζοντας σαν κλειδί το αναγνωριστικό της κάθε ακμής (π.χ. edge14) του αρχείου SVG και σαν τιμή, την ιδιότητα που αντιστοιχεί σε αυτήν την ακμή (π.χ. key: edge32 -> value: rdfs:label).

- public static HashMap initializeLiteralMap(String file)

Στο παραγόμενο SVG αρχείο, σε κάθε λεκτικό αντιστοιχεί ένα μοναδικό αναγνωριστικό. Το αναγνωριστικό αυτό, αντιστοιχεί στην τιμή node και ενός μοναδικού αριθμού για αυτό το λεκτικό (π.χ. node1). Στην συγκεκριμένη μέθοδο επιστρέφουμε ένα αντικείμενο τύπου HashMap, ορίζοντας σαν κλειδί το αναγνωριστικό κάθε λεκτικού (π.χ. node1) του αρχείου SVG και σαν τιμή, την ονομασία του λεκτικού (π.χ. key:literal_6 -> value: Nikololaos Niniadis).

- public String getLitNodesId(String File)

Στην συγκεκριμένη μέθοδο επιστρέφουμε ένα αντικείμενο τύπου String, στο οποίο έχουμε ορίσει δύο JavaScript πινάκες, ένας αφορά τους κόμβους που αντιστοιχούν στα λεκτικά του SVG αρχείου και ένας πίνακας που αντιστοιχεί στα αναγνωριστικά των λεκτικών. Στην συνέχεια εμφανίζεται ένα παράδειγμα του επιστρεφόμενου αντικείμενου String:

```
var nodes=new Array("node25","node27","node11","node15");

var literals=new Array("Literal_6","Literal_7","Literal_4","Literal_5");
```

Οι συγκεκριμένοι πίνακες χρησιμοποιούνται αργότερα από τις JavaScript συναρτήσεις, για την επίτευξη επιλογής των ασαφών ιδιοτήτων και των λεκτικών που αναφέρονται στις ασαφείς ιδίοτητες.

- public String checkLiteralHashMap(String literal, HashMap hashMap)

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο updateSvgFile(String File, String nFile) της ιδία κλάσης. Δοθέντος ενός συγκεκριμένου αναγνωριστικού ενός λεκτικού, καθώς επίσης και του αντικείμενου HashMap που προκύπτει από την κλήση της μεθόδου initialiseLiteralMap(String file), επιστρέφει την ονομασία του λεκτικού που αντιστοιχεί στο αναγνωριστικό.

- public String checkHashMap(String strline, HashMap hashMap)

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο updateSvgFile(String File, String nFile) της ιδία κλάσης. Δοθέντος ενός αντικειμένου String, αρχικά γίνεται διαχωρισμός των

στοιχείων του αντικείμενου, έτσι ώστε να απομονωθεί το αναγνωριστικό της ακμής (π.χ. edge 15) που περιέχεται στην τιμή του αντικειμένου και στην συνεχεία γίνεται αναζήτηση στο αντικείμενο HashMap που προκύπτει από την κλήση της μεθόδου initialiseMap(String file) και επιστρέφεται η ιδιότητα ή σχέση που αντιστοιχεί στο αναγνωριστικό.

- public void updateSvgFile(String File, String nFile)

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο generateGraph(PrintWriter pw, File dotFile, String graphFormat,boolean saveDOTFile) της κλάσης Ontovisualize.java. Ουσιαστικά είναι η μέθοδος που δημιουργεί το αρχείο XHTML το οποίο ορίζεται σαν πηγή στο component step1Frame της index.zul σελίδας. Η μεταβλητή File αντιστοιχεί στην ονομασία του SVG αρχείου και η μεταβλητή nFile στην ονομασία του παραγόμενου αρχείου XHTML. Η συγκεκριμένη μέθοδος επεξεργάζεται καταλλήλως το SVG αρχείο, έτσι ώστε να επιτρέπεται η διαπεραστικότητα με τον χρήστη. Επίσης ενσωματώνονται οι JavaScript συναρτήσεις που έχουν οριστεί σε αντικείμενα String στην κλάση Functions. Μέσα στις συγκεκριμένες JavaScript συναρτήσεις, έχει οριστεί και η Ajax μέθοδος κλήσης του Servlet Step1Servlet.java

7.3.13 *UPDATESVGFILESTEP2A.JAVA*

Περιγραφή Λειτουργίας:

Πρόκειται για μία από τις κλάσεις, η οποία επεξεργάζεται κατάλληλα το SVG αρχείο που αναπαριστά τον RDF γράφο, ενσωματώνοντας επίσης και JavaScript συναρτήσεις, έτσι ώστε να επιτρέπεται η διαδραστικότητα με τον χρήστη. Ουσιαστικά μέσα από τις μεθόδους της συγκεκριμένης κλάσης παράγεται ένα νέο XHTML αρχείο, το οποίο απεικονίζει τις ασαφείς σχέσεις που είχε επιλέξει ο χρήστης, σε προηγούμενο βήμα. Το παραγόμενο αρχείο ορίζεται σαν πηγή στο component step2aFrame της index.zul σελίδας.

Κλήσεις:

```
import java.util.*;
import java.io.*;
```

Μέθοδοι:

- public HashMap initializeMap(String file)

Ακριβώς ιδία λειτουργικότητα με την αντίστοιχη μέθοδο της κλάσης UpdateSvgFileStep1 που περιγράψαμε παραπάνω. Η συγκεκριμένη μέθοδος καλείται από την μέθοδο updateSvgFile(String File, String nFile, Vector selections) της ιδίας κλάσης.

- public String checkHashMap(String strline, HashMap hashMap,)

Ακριβώς ιδία λειτουργικότητα με την αντίστοιχη μέθοδο της κλάσης UpdateSvgFileStep1 που περιγράψαμε παραπάνω. Η συγκεκριμένη μέθοδος καλείται από την μέθοδο updateSvgFile(String File, String nFile, Vector selections) της ιδίας κλάσης.

- public void updateSvgFile(String File, String nFile, Vector selections)

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο editSelection(Vector selections) της κλάσης ConstMainWin.java. Ουσιαστικά είναι η μέθοδος που δημιουργεί το αρχείο XHTML το οποίο ορίζεται σαν πηγή στο component step2aFrame της index.zul σελίδας. Η μεταβλητή File αντιστοιχεί στην ονομασία του SVG αρχείου και η μεταβλητή nFile στην ονομασία του παραγόμενου αρχείου XHTML. Η συγκεκριμένη μέθοδος επεξεργάζεται καταλλήλως το SVG αρχείο, έτσι ώστε να επιτρέπεται η επιλογή των σχέσεων(προκύπτει από αντικείμενο selections) που είχε επιλέξει ο χρήστης σε προηγούμενο βήμα, έτσι ώστε να επιτυγχάνεται η κλήση του Servlet Step2aServlet.java μέσω των συναρτήσεων JavaScript. Οι JavaScript συναρτήσεις ενσωματώνονται με τα αντικείμενα String που έχουν οριστεί στην κλάση Functions. Μέσα στις συγκεκριμένες JavaScript συναρτήσεις, έχει οριστεί και η Ajax μέθοδος κλήσης του Servlet Step2aServlet.java.

7.3.14 *UPDATESVGFILESTEP2B.JAVA*

Περιγραφή Λειτουργίας:

Πρόκειται για μία από τις κλάσεις, η οποία επεξεργάζεται κατάλληλα το SVG αρχείο που αναπαριστά τον RDF γράφο, ενσωματώνοντας επίσης και JavaScript συναρτήσεις, έτσι ώστε να επιτρέπεται η διαδραστικότητα με τον χρήστη. Ουσιαστικά μέσα από τις μεθόδους της συγκεκριμένης κλάσης παράγεται ένα νέο XHTML αρχείο, το οποίο απεικονίζει τις ασαφείς έννοιες που είχε επιλέξει ο χρήστης, σε προηγούμενο βήμα. Το παραγόμενο αρχείο ορίζεται σαν πηγή στο component step2bFrame της index.zul σελίδας.

Κλήσεις:

```
import java.util.*;
import java.io.*;
```

Μέθοδοι:

- public void updateSvgFile(String File, String nFile, Vector selections)

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο initializeListboxesStep2b της κλάσης ConstMainWin.java. Ουσιαστικά είναι η μέθοδος που δημιουργεί το αρχείο XHTML το οποίο ορίζεται σαν πηγή στο component step2bFrame της index.zul σελίδας. Η μεταβλητή File αντιστοιχεί στην ονομασία του SVG αρχείου και η μεταβλητή nFile στην

ονομασία του παραγόμενου αρχείου XHTML. Η συγκεκριμένη μέθοδος επεξεργάζεται καταλλήλως το SVG αρχείο, έτσι ώστε να επιτρέπεται η επιλογή των εννοιών(προκύπτει από αντικείμενο selections) που είχε επιλέξει ο χρήστης σε προηγούμενο βήμα, έτσι ώστε να επιτυγχάνεται η κλήση του Servlet Step2bServlet.java μέσω των συναρτήσεων JavaScript. Οι JavaScript συναρτήσεις ενσωματώνονται με τα αντικείμενα String που έχουν οριστεί στην κλάση Functions. Μέσα στις συγκεκριμένες JavaScript συναρτήσεις, έχει οριστεί και η Ajax μέθοδος κλήσης του Servlet Step2bServlet.java.

7.3.15 *UPDATESVGFILESTEP2C.JAVA*

Περιγραφή Λειτουργίας:

Πρόκειται για μία από τις κλάσεις, η οποία επεξεργάζεται κατάλληλα το SVG αρχείο που αναπαριστά τον RDF γράφο, ενσωματώνοντας επίσης και JavaScript συναρτήσεις, έτσι ώστε να επιτρέπεται η διαδραστικότητα με τον χρήστη. Ουσιαστικά μέσα από τις μεθόδους της συγκεκριμένης κλάσης παράγεται ένα νέο XHTML αρχείο, το οποίο απεικονίζει τις ασαφείς ιδιότητες σε συνδυασμό με τα λεκτικά που αποτελούν τιμές τους, τις οποίες είχε επιλέξει ο χρήστης, σε προηγούμενο βήμα. Το παραγόμενο αρχείο ορίζεται σαν πηγή στο component step2cFrame της index.zul σελίδας.

Κλήσεις:

```
import java.util.*;
import java.io.*;
```

Μέθοδοι:

- public HashMap initializeMap(String file)

Ακριβώς ιδία λειτουργικότητα με την αντίστοιχη μέθοδο της κλάσης UpdateSvgFileStep1 που περιγράψαμε παραπάνω. Η συγκεκριμένη μέθοδος καλείται από την μέθοδο updateSvgFile(String File, String nFile, Vector selections, Vector Literals) της ιδίας κλάσης.

- public String checkHashMap(String strline, HashMap hashMap,)

Ακριβώς ιδία λειτουργικότητα με την αντίστοιχη μέθοδο της κλάσης UpdateSvgFileStep1 που περιγράψαμε παραπάνω. Η συγκεκριμένη μέθοδος καλείται από την μέθοδο updateSvgFile(String File, String nFile, Vector selections, Vector Literals) της ιδίας κλάσης.

- public static String getLiteralUri(String file, String litname)

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο initializeListboxesStep2c () της κλάσης ConstMainWin.java. Δοθέντος ενός αρχείου SVG και ενός αναγνωριστικού ενός λεκτικού (π.χ. Literal_2) επίτρεφει το λεκτικό που αντιστοιχεί σε αυτό το αναγνωριστικό.

- public void updateSvgFile(String File, String nFile, Vector selections, Vector Literals)

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο initializeListboxesStep2c () της κλάσης ConstMainWin.java. Ουσιαστικά είναι η μέθοδος που δημιουργεί το αρχείο XHTML το οποίο ορίζεται σαν πηγή στο component step2cFrame της index.zul σελίδας. Η μεταβλητή File αντιστοιχεί στην ονομασία του SVG αρχείου και η μεταβλητή nFile στην ονομασία του παραγόμενου αρχείου XHTML. Η συγκεκριμένη μέθοδος επεξεργάζεται καταλλήλως το SVG αρχείο, έτσι ώστε να επιτρέπεται η επιλογή των ιδιοτήτων(προκύπτει από το αντικείμενο selections) και των τιμών τους που είναι λεκτικά(προκύπτει από το αντικείμενο Literals), που είχε επιλέξει σε προηγούμενο βήμα ο χρήστης, έτσι ώστε να επιτυγχάνεται η κλήση του Servlet Step2cServlet.java μέσω των συναρτήσεων JavaScript. Οι JavaScript συναρτήσεις ενσωματώνονται με τα αντικείμενα String που έχουν οριστεί στην κλάση Functions. Μέσα στις συγκεκριμένες JavaScript συναρτήσεις, έχει οριστεί και η Ajax μέθοδος κλήσης του Servlet Step2cServlet.java.

7.3.16 *SETSVGSELECTIONS.JAVA*

Περιγραφή Λειτουργίας:

Πρόκειται για την κλάση, η οποία σχετίζεται με τις επιλογές ασαφών στοιχείων που έχει πραγματοποιήσει ο χρήστης σε προγενέστερη χρονική περίοδο. Μέσω των μεθόδων της συγκεκριμένη κλάσης επιτυγχάνεται η απεικόνιση των επιλογών των ασαφών στοιχείων, που είχε επιλέξει ο χρήστης προγενέστερα στο βήμα step1. Οι μέθοδοι της συγκεκριμένης κλάσης, καλούνται μόνο εφόσον πρώτα έχει διαπιστωθεί ότι ο χρήστης πραγματοποιήσει επιλογές προγενέστερα και στην συνέχεια, η εκτέλεση των μεθόδων της αναφερόμενης κλάσης, έχει ως αποτέλεσμα την δημιουργία ενός νέου αρχείου XHTML. Το παραγόμενο αρχείο ορίζεται σαν πηγή στο component step1Frame της index.zul σελίδας.

Κλήσεις:

```
import java.util.*;
import java.io.*;
```

Μέθοδοι:

- public HashMap initializeMap(String file)

Ακριβώς ιδία λειτουργικότητα με την αντίστοιχη μέθοδο της κλάσης UpdateSvgFileStep1 που περιγράψαμε παραπάνω. Η συγκεκριμένη μέθοδος καλείται από την μέθοδο updateSvgFile(String File, String nFile, Vector selections, String values,Vector Literals) της ιδίας κλάσης.

- public String checkHashMap(String strline, HashMap hashMap,)

Ακριβώς ιδία λειτουργικότητα με την αντίστοιχη μέθοδο της κλάσης UpdateSvgFileStep1 που περιγράψαμε παραπάνω. Η συγκεκριμένη μέθοδος καλείται από την μέθοδο updateSvgFile(String File, String nFile, Vector selections, String values,Vector Literals) της ιδίας κλάσης.

- public void updateSvgFile(String File, String nFile, Vector selections, String values, Vector Literals)

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο generateGraph(PrintWriter pw, File dotFile, String graphFormat, boolean saveDOTFile) της κλάσης Ontovisualize.java. Καλείται μόνο εφόσον υπάρχει εγγραφή στο αντικείμενο xmlData τύπου HashMap, με κλειδί την τιμή step1. Η μεταβλητή File αντιστοιχεί στην ονομασία του XHTML αρχείου, το οποίο παράγεται με την εκτέλεση της μεθόδου UpdateSvgFile(String File, String nFile) της κλάσης UpdateSvgFileStep1 και η μεταβλητή nFile στην ονομασία του παραγόμενου αρχείου XHTML. Το αντικείμενο selections αντιστοιχεί στις επιλογές ασαφών στοιχείων που εμπεριέχονται σε μία crisp οντολογία και που είχε πραγματοποιήσει ο χρήστης σε προγενέστερη περίοδο. Το αντικείμενο values τύπου String, αντιστοιχεί σε τρείς πίνακες JavaScript, που προκύπτουν από τις επιλογές του χρήστη καθώς επίσης και σε μια JavaScript συνάρτηση με την ονομασία setData(), η οποία ορίζεται να εκτελεστεί με το που κληθεί το παραγόμενο αρχείο XHTML. Η κλήση της συγκεκριμένης JavaScript συνάρτησης έχει ως αποτέλεσμα την δημιουργία εγγραφών στους HTML πίνακες της παραγόμενου αρχείου, που αναφέρονται στα επιλεγμένα στοιχεία. Τέλος το αντικείμενο Literals εμπεριέχει τα αναγνωριστικά των λεκτικών του SVG αρχείου που ενσωματώνεται στο XHTML αρχείο file (π.χ. Literal_3) και που αποτελούν τιμές στις επιλεγμένες ιδιότητες της Crisp οντολογίας. Η παραπάνω μέθοδος επεξεργάζεται καταλλήλως το XHTML αρχείο, έτσι ώστε να απεικονίζεται με διαφορετική μορφοποίηση στο βήμα step1, τα επιλεγμένα ασαφή στοιχεία.

7.3.17 *ONTOVISUALIZE.JAVA*

Περιγραφή Λειτουργίας:

Η OntoVisualize.java κλάση χρησιμοποιείται για την παραγωγή του RDF γράφου της crisp οντολογίας που έχει οριστεί σε μορφή SVG. Η συγκεκριμένη κλάση έχει βασιστεί κυρίως στους μεθόδους του servlet «ARPServlet.java», μέσω των οποίων επιτυγχάνεται η λειτουργικότητα του RDFValidator[38] της W3C. Οι μέθοδοι που προέρχονται από το servlet ARPServlet.java, έχουν τροποποιηθεί και επεξεργαστεί με τέτοιο τρόπο, έτσι ώστε να προσαρμοστούν στις ανάγκες λειτουργίας της πλατφόρμας «*wb*GraphFuzzyOnto».

Θα πρέπει να επισημάνουμε στο σημείο αυτό, ότι μέσα από τους μεθόδους της κλάσης OntoVisualize χρησιμοποιούνται οι απαραίτητες βιβλιοθήκες του Graph Visualization, για την παραγωγή από ένα αρχείο DOT του RDF γράφου σε μορφή SVG.

Κλήσεις:

```
import org.apache.regexp.RE;
import org.apache.regexp.RESyntaxException;
```

```
import org.xml.sax.ErrorHandler;
import org.xml.sax.SAXParseException;
import org.xml.sax.SAXException;
import org.xml.sax.InputSource;
import org.w3c.dom.*;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.DocumentBuilder;
import java.io.*;
import java.util.*;
import com.hp.hpl.jena.rdf.arp.*;
import com.hp.hpl.jena.rdf.model.*;
import zul.update.UpdateSvgFileStep1;
import zul.onload.SetSvgSelections;
import zul.ConstMainWin;
```

Μεταβλητές Κλάσης:

```
  ->Μεταβλητές που καθορίζουν την εμφάνιση των κόμβων και των ακμών του SVG αρχείου
  private static final String DEFAULT_NODE_COLOR = "black";
  private static final String DEFAULT_NODE_TEXT_COLOR = "blue";
  private static final String DEFAULT_EDGE_COLOR = "darkgreen";
  private static final String DEFAULT_EDGE_TEXT_COLOR = "red";
  private static final String DEFAULT_ORIENTATION = "LR";
  private static final String DEFAULT_FONT_SIZE = "9px";
  private static final String DEFAULT_FONT = "sans-serif";

  ->Τύπος αρχείου που επιθυμούμε να εξάγουμε από ένα .DOT αρχείο
  private static final String FORMAT_SVG_EMBED = "SVG_EMBED";

  -> Αντιστοιχεί στον φάκελο οπού αποθηκεύονται τα αρχεία XHTML και SVG
  protected static String m_ServletTmpDir = null;

 ->Αντιστοιχεί στον φάκελο root του Graph Visualization
  private static String m_GraphVizPath = null;

  ->Αντιστοιχεί στον φάκελο Fonts του Graph VIsualization
  private static String m_GraphVizFontDir = null;

  ->Μεταβλητή που απαιτείται από το Graph Visualization
    private static String DOTFONTPATH = "DOTFONTPATH";

  ->Ονόματα για την δημιουργία προσωρινών αρχείων XHTML και SVG
  private static final String TMP_FILE_PREFIX = "servlet_";
  private static final String SUFFIX_DOT = ".dot";

 ->Κατάληξη SVG αρχείου η όποια χρησιμοποιείται και από το Graph Visualization
  private static final String NAME_SVG = "svg";

  -> Όνομα για το προσωρινό αρχείο DOT
  private static final String DOT_TITLE = "dotfile";

  -> Πρόθεμα που χρησιμοποιείται στους ανώνυμους κόμβους.
  private static final String ANON_NODE = "genid:";

  ->Namespace που χρησιμοποιείται εφόσον δεν έχει οριστεί κανένα στο RDF εγγραφό
  private static final String DEFAULT_NAMESPACE = "http://www.w3.org/RDF/";

  ->Boolean  μεταβλητή που χρησιμοποιείται για να επισημάνει αν ένα RDF έγγραφο
  ->περιλαμβάνει τουλάχιστον μία RDF πρόταση
```

```
static boolean AT_LEAST_ONE_TRIPLE = false;

->Αντικείμενο HashMap για την αποθήκευση των Qnames ενός RDF έγγραφου
public static HashMap prefixes;

->Αντικείμενο HashMap που αντιστοιχεί ένα URI με μία συντόμευση,
->σχετιζόμενα με τα Qnames του RDF γράφου
public static HashMap prefixes_uri;
```

Μέθοδοι:

- private File createTempFile(String directory, String prefix, String suffix)

Πρόκειται για την μέθοδο η οποία δημιουργεί ένα προσωρινό αρχείο DOT στο οποίο περιγράφονται οι RDF προτάσεις του RDF έγγραφου. Η τιμή της μεταβλητής directory αντιστοιχεί στην τοποθεσία του φακέλου, οπού θα δημιουργηθεί το αρχείο. Η τιμή της μεταβλητής prefix αντιστοιχεί στο πρόθεμα του αρχείου, ενώ η τιμή της μεταβλητή suffix στην κατάληξη του αρχείου(.dot). Η συγκεκριμένη μέθοδος καλείται από την generateGraph(PrintWriter pw, File dotFile, boolean saveDOTFile) μέθοδο της ίδιας κλάσης.

- private boolean generateGraphFile(String dotFileName, String outputFileName)

Πρόκειται για την μέθοδο οπού παράγει το SVG αρχείο δοθέντος του DOT αρχείου. Η τιμή της μεταβλητή dotFileName αντιστοιχεί στο αρχείο DOT, ενώ η τιμή της μεταβλητής outputFileName αντιστοιχεί στο αρχείο SVG που πρόκειται να παραχθεί. Η παραγωγή του SVG αρχείου επιτυγχάνεται κάνοντας χρήση του Graph Visualization. Η συγκεκριμένη μέθοδος καλείται από την generateGraph(PrintWriter pw, File dotFile, boolean saveDOTFile) μέθοδο της ίδιας κλάσης.

- private void processGraphParameters(PrintWriter pw)

Πρόκειται για την μέθοδο η όποια προσθέτει επικεφαλίδα (Header) σε ένα υπό παραγωγή αρχείο DOT. Συγκεκριμένα ορίζει σε ένα αντικείμενο PrintWriter από το οποίο θα προκύψει το DOT αρχείο, την επικεφαλίδα του DOT αρχείου. Ορίζει τις μεταβλητές που έχουν κα κάνουν με την μορφοποίηση του RDF γράφου, των ακμών (edges) και των κόμβων(nodes) του. Η συγκεκριμένη μέθοδος καλείται από την generateGraph(PrintWriter pw, File dotFile, boolean saveDOTFile) της ίδιας κλάσης.

- private void clearDirectory(String Directory)

Η συγκεκριμένη μέθοδος διαγραφεί όλα τα αρχεία XHTML ή SVG, στα οποία η ημερομηνίας τροποποίησης τους δεν είναι προγενέστερη της χρονικής περιόδου της μίας ημέρας, από την στιγμή που καλείται η μέθοδος clearDirectory(String Directory). Η τιμής της μεταβλητής Directory αντιστοιχεί στο φάκελο στον οποίο αποθηκεύονται τα αρχεία XHTML ή SVG. Η παραπάνω μέθοδος καλείται από την μέθοδο createDotFile(String srdf, String format) της ιδίας κλάσης.

- private Vector getStep1Data()

Πρόκειται για την μέθοδο η οποία ελέγχει αν ο χρήστης έχει προβεί στην επιλογή ασαφών στοιχείων, σε προγενέστερη χρονική περίοδο. Αρχικά πραγματοποιείται έλεγχος στο αντικείμενο xmlData (τύπου HashMap) της κλάσης ConstMainWin.java. Εξετάζεται αν υπάρχει εγγραφή με κλειδί την τιμή «step1». Σε περίπτωση που υπάρχει, διαχειρίζεται καταλλήλως την τιμή της συγκεκριμένης εγράφης και επιστρέφει ένα αντικείμενο τύπου Vector, που αντιστοιχεί στις επιλογές των ασαφών στοιχείων που είχε πραγματοποιήσει ο χρήστης σε προγενέστερη περίοδο. Η παραπάνω μέθοδος καλείται από την Vector generateGraph(PrintWriter pw, File dotFile, boolean saveDOTFile) της ιδίας κλάσης.

- public static String getCharacterDataFromElement(Element e)

Δοθέντος ενός XML στοιχείου (Element), επιστρέφει την τιμή που περικλείεται μέσα σε αυτό το στοιχείο, εφόσον η τιμή αυτή είναι στιγμιότυπο τύπου «CharacterData». Η συγκεκριμένη μέθοδος καλείται από την μέθοδο getStep1Data () της ίδιας κλάσης.

- public Vector generateGraph(PrintWriter pw, File dotFile, boolean saveDOTFile)

Πρόκειται για την μέθοδο στην οποία ολοκληρώνεται το αρχείο DOT, ορίζοντας τις απαραίτητες τιμές στο αντικείμενο pw, έτσι ώστε στην συνεχεία να μπορεί να διαβαστεί από το Graph Visualization για την παραγωγή του SVG αρχείου. Η δημιουργία του SVG αρχείου επιτυγχάνεται με την κλήση της μεθόδου generateGraphFile(String dotFileName, String outputFileName), την οποία περιγράψαμε παραπάνω. Στην συνεχεία, πραγματοποιείται η δημιουργία του XHTML αρχείου καλώντας την μέθοδο updateSvgFile() της κλάσης UpdateSvgFileStep1(String File, String nFile) και υπό προϋποθέσεις καλείται και η μέθοδος updateSvgFile(String File, String nFile, Vector selections, String values,Vector Literals) της κλάσης SetSvgSelections. Για να κληθεί η μέθοδος της κλάσης SetSvgSelections, θα πρέπει ο χρήστης να έχει προβεί στην επιλογή των ασαφών στοιχειών της συγκεκριμένης οντολογίας, σε προγενέστερη περίοδο. Η μέθοδος generateGraph ολοκληρώνεται με την επιστροφή ενός αντικείμενου τύπου Vector, το οποίο εμπεριέχει δυο έγγραφες τύπου String. Η πρώτη εγγραφή εμπεριέχει την τιμή «success» ή «failure» ανάλογα με την έκβαση της διαδικασίας δημιουργίας του SVG αρχείου. Η δεύτερη εγγραφή θα περιέχει το όνομα του παραγόμενου αρχείου, εφόσον η όλη διαδικασία ήταν επιτυχής ή το μήνυμα λάθους εφόσον η διαδικασία δημιουργίας του SVG αρχείου παρουσίασε σφάλμα.

- public static String replaceString(String input, String key, String replacement)

Πρόκειται για μία βοηθητική μέθοδο, η οποία πραγματοποιεί αναζήτηση σε ένα αντικείμενο τύπου String (input), βάση μιας συγκεκριμένης συμβολοσειράς(key). Εφόσον υπάρχει η συγκεκριμένη συμβολοσειρά, πραγματοποιείται η αντικατάσταση της με ένα συγκεκριμένο αντικείμενο τύπου String(replacement).

- public static void initialize()

Μέσα από την συγκεκριμένη κλάση αρχικοποιούνται οι ακόλουθες μεταβλητές της κλάσης:

- m_ServletTmpDir
- m_GraphVizPath
- m_GraphVizFontDir
- prefixes
- prefixes_uri

Η συγκεκριμένη μέθοδος καλείται από την μέθοδο createDotFile(String srdf, String format) της ιδίας κλάσης.

- public void getPrefixes(String srdf)

Δοθέντος ενός RDF έγγραφου, η συγκεκριμένη κλάση βρίσκει τα Qnames που έχουν οριστεί στο RDF έγγραφο και δημιουργεί εγγραφές στο αντικείμενο prefixes, τοποθετώντας σε κάθε εγράφη του σαν κλειδί το πρόθεμα του URI(π.χ. rdf) και σαν τιμή το URI(http://www.w3.org/1999/02/22-rdf-syntax-ns). Η συγκεκριμένη μέθοδος καλείται από την μέθοδο createDotFile(String srdf, String format) της ιδίας κλάσης.

- public static String getPrefixWithURI(String URI)

Πρόκειται για μία μέθοδο η οποία χρησιμοποιείται κατά την δημιουργία του DOT αρχείου, για την μετατροπή ενός URI που αντιστοιχεί σε ένα στοιχείο της RDF πρότασης, σε μορφή συντομεύσεως κάνοντας χρήση των Qnames που μπορεί να έχουν οριστεί στην αρχή του RDF έγγραφου. Επίσης μέσω της συγκεκριμένης μεθόδου προσθέτονται εγγραφές στο αντικείμενο prefixes_uri.

- public static String getUrifromPrefix_Uri(String pref_uri)

Πρόκειται για μία μέθοδο η οποία χρησιμοποιείται από μεθόδους της κλάσης ConstMainWin.java. Δοθέντος ενός URI σε μορφή συντομεύσεως(π.χ. rdf_:Team_Class0) επιστρέφει το URI στην κανονική του μορφή(π.χ. http://protege.stanford.edu/rdf/Team_Class0).

- public Vector createDotFile(String srdf, String format)

Πρόκειται για την μέθοδο, μέσω της οποία ξεκινάει η διαδικασία παράγωγης του XHTML αρχείου απεικόνισης του RDF γράφου, που ορίζεται ως πηγή στο component step1Frame της index.zul σελίδας. Μέσα από την συγκεκριμένη μέθοδο δημιουργούνται τα απαραίτητα αντικείμενα PrintWriter, StringWriter και OutputStreamWriter που απαιτούνται για την δημιουργία του DOT αρχείου, αλλά και της δυνατότητας εγγραφής σε αυτό. Στην συνέχεια δημιουργείται ένα αντικείμενο τύπου SH μέσω του οποίου θα διαχειριστούν καταλλήλως οι RDF προτάσεις που απαρτίζουν το RDF έγγραφο και να εκφραστούν στην συνεχεία στο DOT αρχείο. Μετά την δημιουργία του αντικείμενου SH (που πρόκειται ουσιαστικά για ένα αντικείμενο τύπου StatementHandler) δημιουργείται ένα αντικείμενο τύπου SaxErrorHandler για την διαχείριση των σφαλμάτων που μπορεί να προκύψουν. Τα δύο αντικείμενα τύπου SH και SaxErrorHandler ορίζονται με τις κατάλληλες μεθόδους

(setStatementHandler(sh) και setErrorHandler(errorHandler) αντίστοιχα) σε ένα αντικείμενο ARP parser. Μέσω ενός αντικείμενου ARP parser μπορούμε να «φορτώσουμε» ένα έγγραφο RDF και να το διαχειριστούμε. Η μέθοδος ολοκληρώνεται με την κλήση της μεθόδου printErrorMessages(SaxErrorHandler eh) μέσω της οποία εμφανίζονται τυχόν μηνύματα λάθους, σε διαφορετική περίπτωση καλείται η μέθοδος generateGraph(PrintWriter pw, File dotFile, boolean saveDOTFile).

- private Vector printErrorMessages(SaxErrorHandler eh)

Πρόκειται για μία μέθοδο η οποία χρησιμοποιείται για να διαχειριστεί καταλλήλως εφόσον υπάρχουν μηνύματα σφάλματος, που προκύπτουν από το αντικείμενο SaxErrorHandler. Σε περίπτωση σφάλματος, επιστρέφει ένα αντικείμενο τύπου Vector, όπου περιέχει δύο αντικείμενα τύπου String με τιμές «failure» και το «μήνυμα λάθους» αντίστοιχα, σε κάθε αντικείμενο String.

7.3.18 *SH.JAVA*

Περιγραφή Λειτουργίας:

Η SH κλάση μια εμφωλευμένη κλάση της OntoVisualize.java κλάσης, η οποία χρησιμοποιείται για την διαχείριση των RDF προτάσεων και για την έκφραση των προτάσεων αυτών με τους καταλλήλους όρους στο DOT αρχείο. Η SH υλοποιεί την διεπαφή StatementHandler.

Μεταβλητές Κλάσης:

```
->Αντιστοιχεί στο αντικείμενο τύπου PrintWriter που χρησιμοποιείται για την εγγραφή του DOT αρχείου
PrintWriter pw;
->Δηλώνει το αν θα τοποθετηθεί κάποια ετικέτα στους κενούς κόμβους του RDF γράφου
boolean anonNodesEmpty;
->Αντιστοιχεί στον αριθμό των προτάσεων του RDF εγγράφου
int numStatements;
->Αντιστοιχεί στον αριθμό των λεκτικών του RDF εγγράφου
int numLiterals;
Hashtable subjects;
->Αντιστοιχεί στον αριθμό των υποκειμένων του RDF εγγράφου
int numSubjects;
->Αντιστοιχεί στο τύπου(format) του αρχείου του γράφου που θέλουμε να δημιουργήσουμε που είναι η -
->τιμή «svg_embeded»
String gFormat;
```

Μέθοδοι:

- public SH(PrintWriter pw, boolean anonNodesEmpty, String graphFormat)

Πρόκειται για τον «Constructor» της συγκεκριμένης κλάσης. Καλείται από την μέθοδο createDotFile(String srdf, String format) της OntoVisualize.java κλάσης.

- public void statement(AResource subj, AResource pred, AResource obj)

Πρόκειται για έναν Handler για μία RDF πρόταση, η οποία αποτελείται από τρείς πόρους. Μέσω της συγκεκριμένης μεθόδου καλείται η μέθοδος statementDotResource(AResource subj, AResource pred, AResource obj) της ίδιας κλάσης.

- public void statement(AResource subj, AResource pred, ALiteral lit)

Πρόκειται για έναν Handler για μία RDF πρόταση, οπού η τιμή της ιδιότητας αντιστοιχεί σε λεκτικό. Μέσω της συγκεκριμένης μεθόδου καλείται η μέθοδος void statementDotLiteral(AResource subj, AResource pred, ALiteral lit) της ίδιας κλάσης. Επίσης αυξάνεται η μεταβλητή numLiterals κάθε φόρα που καλείται η συγκεκριμένη μέθοδος.

- public void printFirstPart(AResource subj)

Μέσω της συγκεκριμένης μεθόδου δημιουργείται το πρώτο μέρος του DOT αρχείου που αφορά μια RDF πρόταση, ανεξαρτήτως αν στο ρόλο του αντικείμενου βρίσκεται ένας πόρος ή ένα λεκτικό. Τέλος μέσω της συγκεκριμένης μεθόδου καλείται και η μέθοδος getPrefixWithURI(String URI) της κλάσης OntoVIsualize.java.

- public void statementDotResource(AResource subj, AResource pred, AResource obj)

Μέσω της συγκεκριμένης μεθόδου περιγράφεται με όρους της DOT γλώσσας, μια RDF πρόταση που αποτελείται από τρείς πόρους.

- Public void statementDotLiteral(AResource subj, AResource pred, ALiteral lit)

Μέσω της συγκεκριμένης μεθόδου, περιγράφεται με όρους της DOT γλώσσας μια RDF πρόταση, οπού τον ρόλο του αντικείμενου σε αυτήν την πρόταση κατέχει ένα λεκτικό.

7.3.19 *SAXERRORHANDLER.JAVA*

Περιγραφή Λειτουργίας:

Η SaxErrorHandler κλάση μια εμφωλευμένη κλάση της OntoVisualize.java κλάσης, η οποία χρησιμοποιείται για την διαχείριση των σφαλμάτων που μπορεί να προκύψουν από την διαχείριση των RDF προτάσεων και την έκφραση των προτάσεων αυτών, με τους καταλλήλους όρους στο DOT αρχείο. Η SH υλοποιεί την διεπαφή ErrorHandler.

Μεταβλητές Κλάσης:

```
->Υλοποιούν τις μεταβλητές που ορίζονται στην διεπαφή   ErrorHandler
String fatalErrors = "";
String errors = "";
String warnings = "";
String datatypeErrors = "";
```

Μέθοδοι:

- public SaxErrorHandler ()

Πρόκειται για τον «Constructor» της συγκεκριμένης κλάσης. Καλείται από την μέθοδο createDotFile() της OntoVisualize.java κλάσης.

- private static String format(SAXParseException e)

Επεξεργάζεται το μήνυμα λάθους που προκύπτει από το αντικείμενο τύπου SAXParseException.

- public void error(SAXParseException e)

Μέσω της συγκεκριμένης μεθόδου ορίζεται τιμή στην μεταβλητή errors. Η τιμή της μεταβλητής errors προκύπτει από το αντικείμενο e τύπου SAXParseException. Στην συγκεκριμένη κλάση γίνεται χρήση της μεθόδου format(SAXParseException e).

- public void fatalError(SAXParseException e)

Μέσω της συγκεκριμένης μεθόδου ορίζεται τιμή στην μεταβλητή fatalErrors. Η τιμή της μεταβλητής fatalErrors προκύπτει από το αντικείμενο e τύπου SAXParseException. Στην συγκεκριμένη κλάση γίνεται χρήση της μεθόδου format(SAXParseException e).

- public void warning (SAXParseException e)

Μέσω της συγκεκριμένης μεθόδου ορίζεται τιμή στην μεταβλητή warnings. Η τιμή της μεταβλητής warnings προκύπτει από το αντικείμενο e τύπου SAXParseException. Στην συγκεκριμένη κλάση γίνεται χρήση της μεθόδου format(SAXParseException e).

- public String getErrors()

Πρόκειται για την μέθοδο «getter» της μεταβλητής errors. Η συγκεκριμένη μέθοδος καλείται από την μέθοδο printErrorMessages(SaxErrorHandler eh) της κλάσης OntoVisualize.java.

- public String getFatalErrors()

Πρόκειται για την μέθοδο «getter» της μεταβλητής fatalErrors. Η συγκεκριμένη μέθοδος καλείται από την μέθοδο printErrorMessages(SaxErrorHandler eh) της κλάσης OntoVisualize.java.

- public String getWarnings()

Πρόκειται για την μέθοδο «getter» της μεταβλητής warnings. Η συγκεκριμένη μέθοδος καλείται από την μέθοδο printErrorMessages(SaxErrorHandler eh) της κλάσης OntoVisualize.java.

- public String getDatatypeErrors()

Πρόκειται για την μέθοδο «getter» της μεταβλητής datatypeErrors.

8 ΠΡΟΣΟΜΟΙΩΣΗΣ ΠΛΑΤΦΟΡΜΑΣ *wb*GRAPHFUZZYONTO

Σκοπός αυτού του κεφαλαίου, είναι να πραγματοποιηθεί μια προσομοίωση της διαδικτυακής πλατφόρμας «*wb*GraphFuzzyOnto», εκτελώντας ένα συγκεκριμένο σενάριο ασαφοποίησης crisp οντολογίας, μέσω του οποίου θα μπορεί ο αναγνώστης να κατανοήσει την λειτουργικότητα των υπηρεσιών που παρέχονται από την σχετική πλατφόρμα.

Αρχικά θα περιγραφτούν οι κλάσεις, οι σχέσεις , οι ιδιότητες και οι τιμές των ιδιοτήτων που απαρτίζουν την crisp οντολογία. Καθώς επίσης και το αντικείμενο που διαπραγματεύεται η οντολογία. Η συγκεκριμένη διαδικασία θεωρείται απαραίτητη, διότι όπως έχουμε ήδη αναφέρει, η λειτουργία της πλατφόρμα «*wb*GraphFuzzyOnto» βασίζεται κυρίως στην μεθοδολογία ασαφοποίησης οντολογιών IKARUS-Onto, η οποία προαπαιτεί την ύπαρξη μιας συμβατικής οντολογίας. Στην συνέχεια θα ξεκινήσει η διαδικασία ασαφοποίησης της οντολογίας, κάνοντας χρήση της υλοποιημένης πλατφόρμας ασαφοποίησης.

8.1 CRISP ΟΝΤΟΛΟΓΙΑ

Το αντικείμενο που διαπραγματεύεται η οντολογία που θα χρησιμοποιηθεί από την πλατφόρμα ασαφοποίησης, αναπαριστά γνώση που σχετίζεται με την περιγραφή «τροχαίων ατυχημάτων». Η ονομασία της συγκεκριμένης οντολογίας είναι «Traffic_Accident». Οι κλάσεις που έχουν υλοποιηθεί είναι οι ακόλουθες :

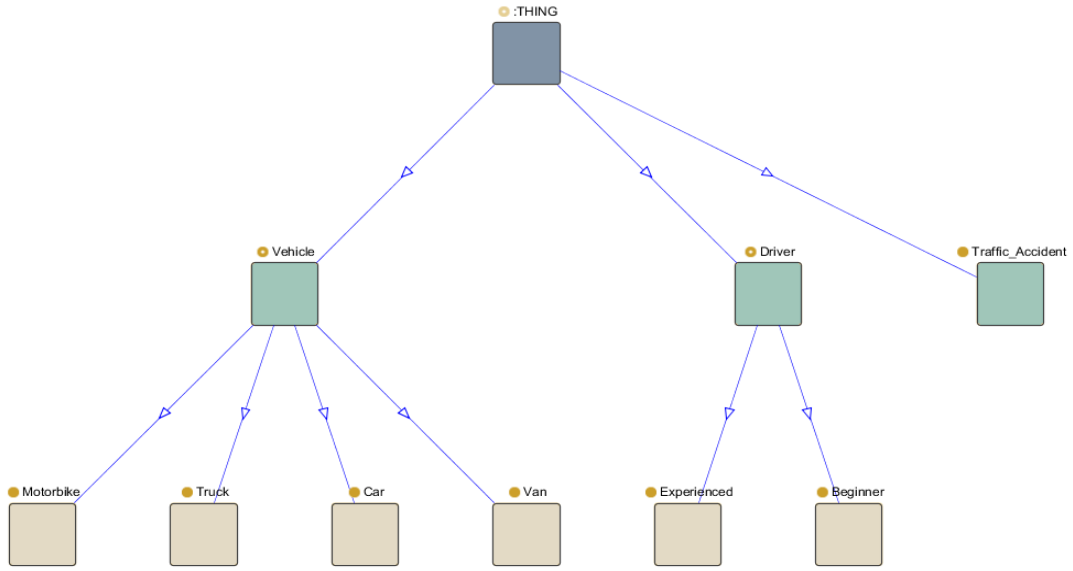
- Driver (Υπερκλάση - Οδηγός)
  - Beginner (Υποκλάση - Αρχάριος)
  - Experienced (Υποκλάση - Έμπειρος)
- Vehicle (Υπερκλάση – Μεταφορικό Μέσο)
  - Car (Υποκλάση- Αυτοκίνητο)
  - Motorbike (Υποκλάση- Μοτοσυκλέτα)
  - Truck (Υποκλάση- Μεγάλο Φορτηγό)
  - Van (Υποκλάση- Μικρό Φορτηγό)

Θα πρέπει να επισημάνουμε κάποια σημεία της οντολογίας για την καλύτερη κατανόηση τους, πριν την επεξήγηση των κλάσεων και των ιδιοτήτων τους:

- Οι διαφορές υποκλάσεις κληρονομούν τις ιδιότητες των υπερκλάσεων τους.
- Όλες οι κλάσεις του ίδιου επιπέδου είναι ανεξάρτητες μεταξύ τους, πράγμα που σημαίνει ότι ένα αντικείμενο δεν μπορεί να ανήκει, ταυτόχρονα, σε πάνω από μία κλάσεις.

- Σε κάθε ιδιότητα μπορεί να ορίζονται κάποια από τα παρακάτω χαρακτηριστικά:
  - Single: μπορεί να λάβει μόνο μία τιμή.
  - Multiple: μπορεί να λάβει πάνω από μία τιμές.
  - String: λαμβάνει αλφαριθμητικά δεδομένα.
  - Date: λαμβάνει τιμές τύπου Ημερομηνίας.
  - Symbol. Σύνολο προκαθορισμένων αλφαριθμητικών τιμών.
  - Instance. Στιγμιότυπο κάποιας κλάσης της οντολογίας. Η ιδιότητα σε αυτήν την περίπτωση αντιστοιχεί σε μία σχέση που συνδέει δυο στιγμιότυπα κλάσεων.

Στην εικόνα 8.1, παρουσιάζεται η δενδρική μορφή ιεραρχίας των κλάσεων της οντολογίας Traffic_Accident, κάνοντας χρήση του εργαλείου «Jambalaya» του προγράμματος ανάπτυξης οντολογιών Protégé 3.4.4 [39].



Εικόνα 8.1 – Δενδρική Αναπαράσταση των κλάσεων της οντολογίας «Traffic_Accident»

8.1.1 *VEHICLE*

Η κλάση Vehicle αντιστοιχεί στην υπερκλάση όλων των μεταφορικών μέσων.

Ιδιότητες:

- **vehicle_number**. Αντιστοιχεί στον αριθμό κυκλοφορίας του αυτοκινήτου και έχει οριστεί να είναι τύπου String και Single.

8.1.2 *CAR*

Η κλάση Car είναι υποκλάση της Vehicle και αντιστοιχεί σε ένα μεταφορικό μέσο τύπου αυτοκίνητου.

Ιδιότητες:

- **vehicle_number**. Κληρονομεί την αντίστοιχη ιδιότητα της υποκλάσης της.

8.1.3 *MOTORBIKE*

Η κλάση Motorbike είναι υποκλάση της Vehicle και αντιστοιχεί σε ένα μεταφορικό μέσο τύπου μοτοσυκλέτας.

Ιδιότητες:

- **vehicle_number**. Κληρονομεί την αντίστοιχη ιδιότητα της υποκλάσης της.

8.1.4 *TRUCK*

Η κλάση Truck είναι υποκλάση της Vehicle και αντιστοιχεί σε ένα μεταφορικό μέσο τύπου μεγάλου φορτηγού.

Ιδιότητες:

- **vehicle_number**. Κληρονομεί την αντίστοιχη ιδιότητα της υποκλάσης της.

8.1.5 *VAN*

Η κλάση Van είναι υποκλάση της Vehicle και αντιστοιχεί σε ένα μεταφορικό μέσο τύπου μικρού φορτηγού.

Ιδιότητες:

- **vehicle_number**. Κληρονομεί την αντίστοιχη ιδιότητα της υποκλάσης της.

8.1.6 *DRIVER*

Η κλάση Driver αναφέρεται στους οδηγούς που συμμετείχαν σε ένα τροχαίο ατύχημα. Η συγκεκριμένη κλάση είναι υπερκλάση των κλάσεων Beginner και Experienced.

Ιδιότητες:

- **aged**. Αναφέρεται στην ηλικία ενός οδηγού. Έχει ορισθεί να είναι τύπου Symbol. Συγκεκριμένα επιτρέπεται η επιλογή μίας εκ των εξής τιμών:

- youth (νέος)
- middle-age (μέσης ηλικίας)
- old (ηλικιωμένος)

- **fathername.** Αντιστοιχεί στο όνομα του πατέρα του οδηγού. Έχει οριστεί να είναι τύπου String και Single.
- **firstname.** Αντιστοιχεί στο όνομα του οδηγού. Έχει οριστεί να είναι τύπου String και Single.
- **lastname.** Αντιστοιχεί στο επίθετο του οδηγού. Έχει οριστεί να είναι τύπου String και Single.
- **licence_number.** Αντιστοιχεί στο αριθμό διπλώματος οδήγησης του κάθε οδηγού, Έχει οριστεί να είναι τύπου String και Single.

Σχέσεις:

- **is_responsible_for**. Αναφέρεται στο γεγονός ότι ένας οδηγός μπορεί να είναι υπεύθυνος για την πρόκληση ενός ατυχήματος. Η τιμή της συγκεκριμένης σχέσης, είναι ένα στιγμιότυπο μιας κλάσης Traffic_Accident.
- **was_driving**. Αναφέρεται στο μέσο μεταφοράς που οδηγούσε ένας οδηγός, που συμμετείχε σε ένα ατύχημα. Η τιμή της συγκεκριμένης σχέσης είναι ένα στιγμιότυπο μιας κλάσης Vehicle και κατά συνέπεια τα στιγμιότυπα των υποκλάσεων της.

8.1.7 *BEGINNER*

Η κλάση Beginner είναι υποκλάση της Driver και αναφέρεται στους αρχάριους οδηγούς.

Επισήμανση:

Κληρονομεί **όλες** τις **ιδιότητες** και **σχέσεις** της κλάσεις Driver.

8.1.8 *EXPERIENCED*

Η κλάση Experienced είναι υποκλάση της Driver και αναφέρεται στους αρχάριους οδηγούς.

Επισήμανση:

Κληρονομεί **όλες** τις **ιδιότητες** και **σχέσεις** της κλάσεις Driver.

8.1.9 *TRAFFIC_ACCIDENT*

Η κλάση Traffic_Accident αναφέρεται σε ένα τροχαίο ατύχημα.

Ιδιότητες:

- **date_happened**. Ημερομηνία που πραγματοποιήθηκε το ατύχημα. Έχει οριστεί να είναι τύπου Date και Single.
- **location**. Στοιχεία σημείου που πραγματοποιήθηκε το ατύχημα. Έχει οριστεί να είναι τύπου String και Single.

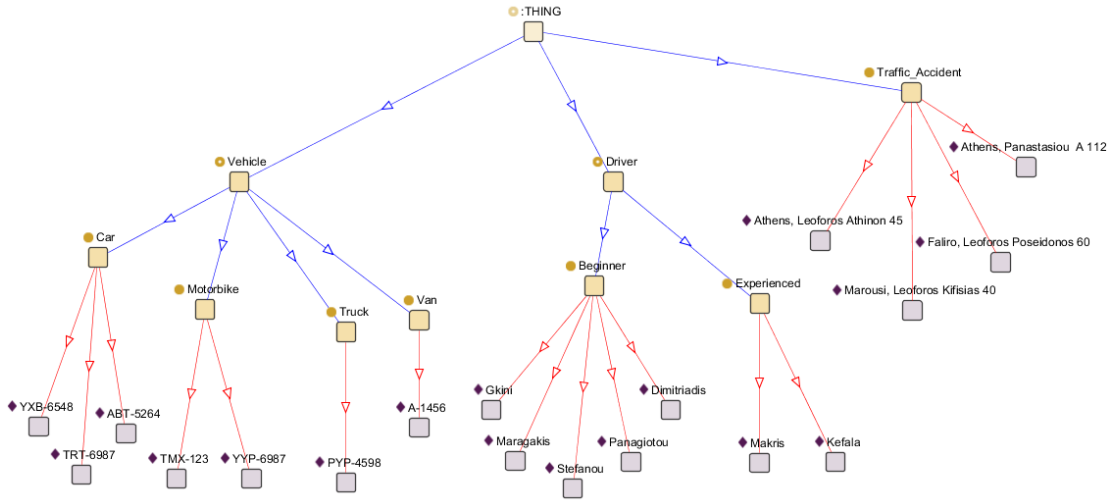
Ολοκληρώνοντας την συγκεκριμένη ενότητα, θα παρουσιάσουμε παρακάτω την οντολογία σε μορφή RDF σημασιολογικής γλώσσας, καθώς επίσης και ένα διάγραμμα απεικόνισης των στιγμιότυπων των παραπάνω κλάσεων σε δενδρική μορφή (εικόνα 8.2).

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE rdf:RDF [
        <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
        <!ENTITY rdf_ 'http://protege.stanford.edu/rdf/'>
        <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
]>
<rdf:RDF xmlns:rdf="&rdf;"
        xmlns:rdf_="&rdf_;"
        xmlns:rdfs="&rdfs;">
<rdf_:Beginner rdf:about="&rdf_;traffic_accident_Class15"      → Ορισμός στιγμιότυπου τύπου Beginner
        rdf_:aged="youth"
        rdf_:fathername="Anastasios"
        rdf_:firstname="Maria"
        rdf_:lastname="Panagiotou"
        rdf_:licence_number="A12654"
        rdfs:label="Panagiotou">
        <rdf_:is_responsible_for rdf:resource="&rdf_;traffic_accident_Class23"/>
        <rdf_:was_driving rdf:resource="&rdf_;traffic_accident_Class30"/>
</rdf_:Beginner>
<rdf_:Experienced rdf:about="&rdf_;traffic_accident_Class16"      → Ορισμός στιγμιότυπου τύπου Experienced
        rdf_:aged="middle-age"
        rdf_:fathername="Georgios"
        rdf_:firstname="Konstantinos"
        rdf_:lastname="Makris"
        rdf_:licence_number="B65412"
        rdfs:label="Makris">
        <rdf_:is_responsible_for rdf:resource="&rdf_;traffic_accident_Class22"/>
        <rdf_:was_driving rdf:resource="&rdf_;traffic_accident_Class29"/>
</rdf_:Experienced>
<rdf_:Beginner rdf:about="&rdf_;traffic_accident_Class17"      → Ορισμός στιγμιότυπου τύπου Beginner
        rdf_:aged="middle-age"
        rdf_:fathername="Dimitrios"
        rdf_:firstname="Alexios"
        rdf_:lastname="Maragakis"
        rdf_:licence_number="C23654"
        rdfs:label="Maragakis">
        <rdf_:is_responsible_for rdf:resource="&rdf_;traffic_accident_Class25"/>
        <rdf_:was_driving rdf:resource="&rdf_;traffic_accident_Class32"/>
</rdf_:Beginner>
<rdf_:Beginner rdf:about="&rdf_;traffic_accident_Class18"      → Ορισμός στιγμιότυπου τύπου Beginner
```

```
            rdf_:aged="old"
            rdf_:fathername="Konstantinos"
            rdf_:firstname="Dimitra"
            rdf_:lastname="Gkini"
            rdf_:licence_number="T60253"
            rdfs:label="Gkini">
           <rdf_:is_responsible_for rdf:resource="&rdf_;traffic_accident_Class22"/>
           <rdf_:was_driving rdf:resource="&rdf_;traffic_accident_Class27"/>
</rdf_:Beginner>
<rdf_:Beginner rdf:about="&rdf_;traffic_accident_Class19"      → Ορισμός στιγμιότυπου τύπου Beginner
            rdf_:aged="middle-age"
            rdf_:fathername="Manolis"
            rdf_:firstname="Ioannis"
            rdf_:lastname="Dimitriadis"
            rdf_:licence_number="T87965"
            rdfs:label="Dimitriadis">
           <rdf_:is_responsible_for rdf:resource="&rdf_;traffic_accident_Class24"/>
           <rdf_:was_driving rdf:resource="&rdf_;traffic_accident_Class33"/>
</rdf_:Beginner>
<rdf_:Beginner rdf:about="&rdf_;traffic_accident_Class20"      → Ορισμός στιγμιότυπου τύπου Beginner
            rdf_:aged="old"
            rdf_:fathername="Panagiotis"
            rdf_:firstname="Katerina"
            rdf_:lastname="Stefanou"
            rdf_:licence_number="N65498"
            rdfs:label="Stefanou">
           <rdf_:is_responsible_for rdf:resource="&rdf_;traffic_accident_Class24"/>
           <rdf_:was_driving rdf:resource="&rdf_;traffic_accident_Class26"/>
</rdf_:Beginner>
<rdf_:Experienced rdf:about="&rdf_;traffic_accident_Class21"      → Ορισμός στιγμιότυπου τύπου Experienced
            rdf_:aged="old"
            rdf_:fathername="Charalambos"
            rdf_:firstname="Anastasia"
            rdf_:lastname="Kefala"
            rdf_:licence_number="K69875"
            rdfs:label="Kefala">
           <rdf_:is_responsible_for rdf:resource="&rdf_;traffic_accident_Class23"/>
           <rdf_:was_driving rdf:resource="&rdf_;traffic_accident_Class28"/>
</rdf_:Experienced>
<rdf_:Traffic_Accident rdf:about="&rdf_;traffic_accident_Class22"→ Ορισμός στιγμιότυπου τύπου Traffic_Accident
            rdf_:date_happened="06/11/2009"
            rdf_:location="Marousi, Leoforos Kifisias 40"
            rdfs:label="Marousi, Leoforos Kifisias 40"/>
<rdf_:Traffic_Accident rdf:about="&rdf_;traffic_accident_Class23"→Ορισμός στιγμιότυπου τύπου Traffic_Accident
            rdf_:date_happened="12/04/2010"
            rdf_:location="Athens, Leoforos Athinon 45"
            rdfs:label="Athens, Leoforos Athinon 45"/>
<rdf_:Traffic_Accident rdf:about="&rdf_;traffic_accident_Class24"→Ορισμός στιγμιότυπου τύπου Traffic_Accident
            rdf_:date_happened="19/05/2010"
            rdf_:location="Faliro, Leoforos Poseidonos 60"
            rdfs:label="Faliro, Leoforos Poseidonos 60"/>
<rdf_:Traffic_Accident rdf:about="&rdf_;traffic_accident_Class25"→ Ορισμός στιγμίοτυπου τύπου Traffic_Accident
            rdf_:date_happened="10/04/2010"
            rdf_:location="Athens, Panastasiou  A 112"
            rdfs:label="Athens, Panastasiou  A 112"/>
<rdf_:Car rdf:about="&rdf_;traffic_accident_Class26"→ Ορισμός στιγμιότυπου τύπου Car
            rdf_:vehicle_number="YXB-6548"
            rdfs:label="YXB-6548"/>
<rdf_:Car rdf:about="&rdf_;traffic_accident_Class27"→ Ορισμός στιγμιότυπου τύπου Car
            rdf_:vehicle_number="TRT-6987"
            rdfs:label="TRT-6987"/>
<rdf_:Van rdf:about="&rdf_;traffic_accident_Class28" → Ορισμός στιγμιότυπου τύπου Van
            rdf_:vehicle_number="A-1456"
```

```
            rdfs:label="A-1456"/>
<rdf_:Truck rdf:about="&rdf_;traffic_accident_Class29"  ➔Ορισμός στιγμιότυπου τύπου Truck
            rdf_:vehicle_number="PYP-4598"
            rdfs:label="PYP-4598"/>
<rdf_:Motorbike rdf:about="&rdf_;traffic_accident_Class30"  ➔ Ορισμός στιγμιότυπου τύπου Motorbike
            rdf_:vehicle_number="TMX-123"
            rdfs:label="TMX-123"/>
<rdf_:Motorbike rdf:about="&rdf_;traffic_accident_Class32"  ➔  Ορισμός στιγμιότυπου τύπου Motorbike
            rdf_:vehicle_number="YYP-6987"
            rdfs:label="YYP-6987"/>
<rdf_:Car rdf:about="&rdf_;traffic_accident_Class33"  ➔ Ορισμός στιγμιότυπου τύπου Car
            rdf_:vehicle_number="ABT-5264"
            rdfs:label="ABT-5264"/>
</rdf:RDF>
```

Πίνακας 8.1- RDF Κωδικας Οντολογίας Traffic_Accident



Εικόνα 8.2 - Αναπαράσταση των στιγμιότυπων των κλάσεων της οντολογίας «Traffic_Accident»

8.2 Επιλογη Ασαφων Στοιχειων

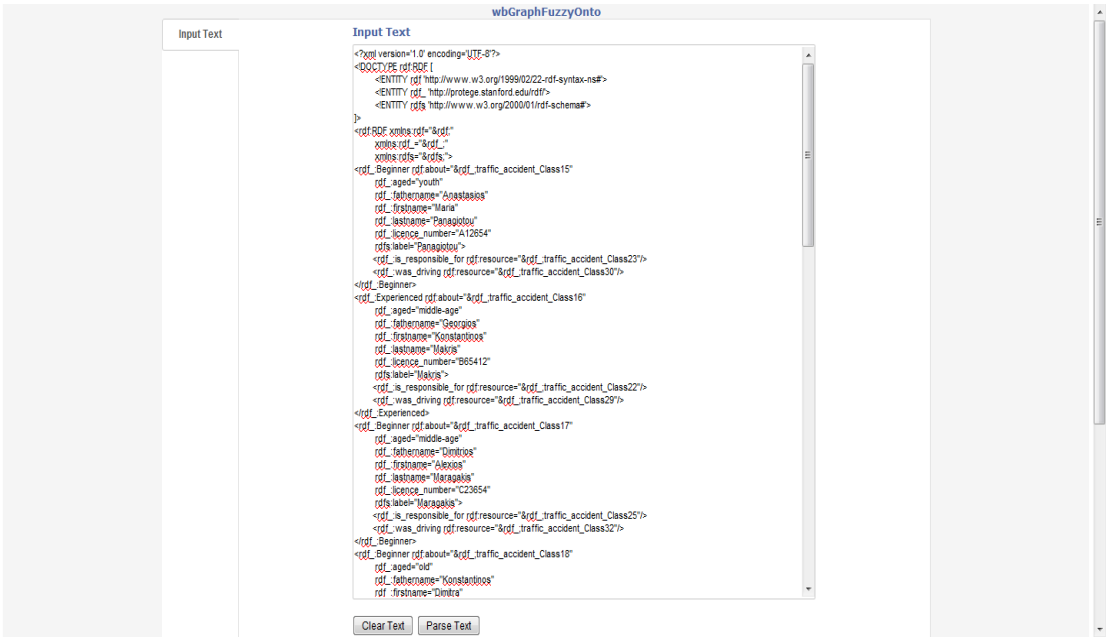
Μετά τον ορισμό της οντολογία «Traffic_Accident» και συγκεκριμένα του RDF έγγραφου της, είμαστε σε θέση να χρησιμοποιήσουμε την πλατφόρμα «*wb*GraphFuzzyOnto» και να προβούμε στην ασαφοποίηση των ασαφών στοιχείων της οντολογίας της «Traffic_Accident».

Στην αρχική σελίδα της πλατφόρμας «*wb*GraphFuzzyOnto» (εικόνα 8.3), εισάγουμε τα στοιχεία της RDF σημασιολογικής γλώσσας που συνθέτουν την οντολογία «Traffic_Accident» και τα οποία παρουσιάσαμε στον πίνακα 8.1. Μόλις ένας χρήστης, που ανήκει στην ομάδα των «Μηχανικών Οντολογιών» επιλέξει το κουμπί με τίτλο «Parse Text», μεταφερόμαστε στο tab «Step1» (εικόνα 8.4). Στο συγκεκριμένο tab απεικονίζεται ο RDF γράφος της οντολογίας «Traffic_Accident». Δίνεται η ευχέρεια λοιπόν, στους μηχανικούς οντολογιών να αποφανθούν για τα στοιχεία τα οποία αποδίδουν ασαφή γνώση, για την

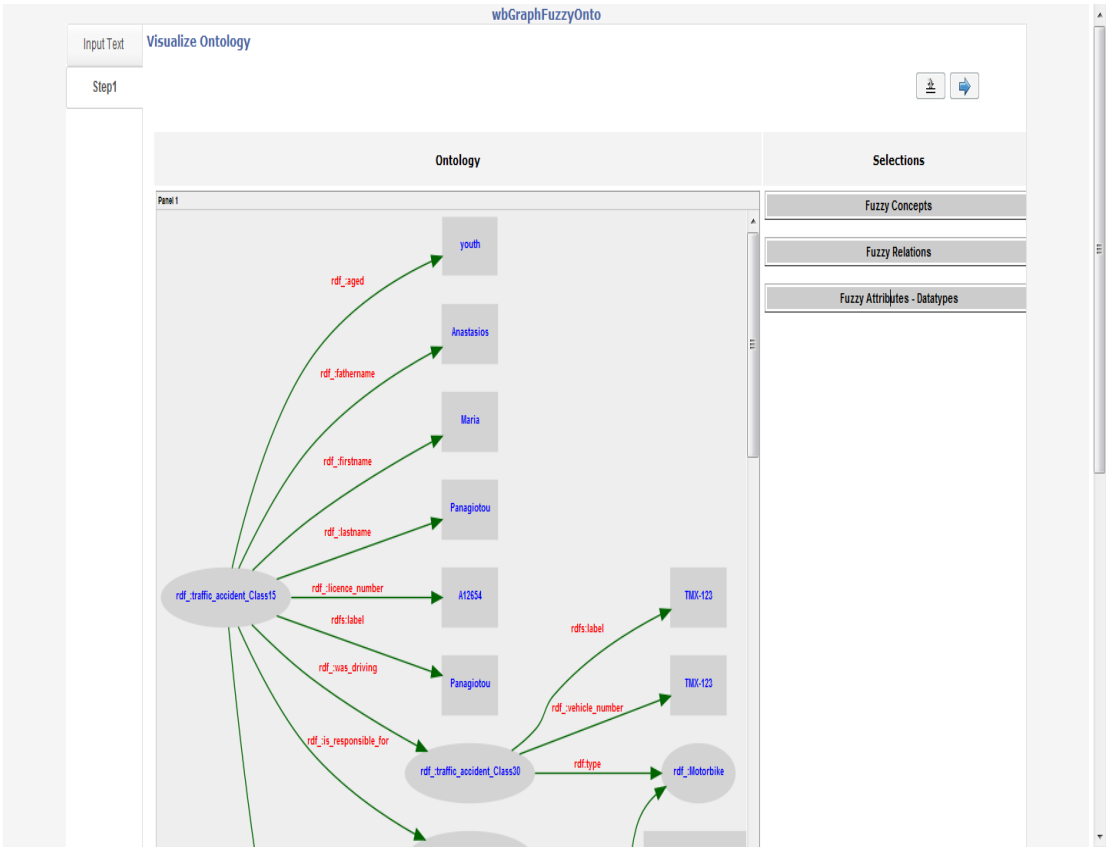
αναγκαιότητα της ασαφοποίησης των στοιχείων αυτών και εφόσον επιθυμούν να συνεχίσουν την ασαφοποίηση τους, τους δίνεται η δυνατότητα επιλογής τους.

Μελετώντας τον RDF γράφο της οντολογίας «Traffic_Accident», παρατηρούμε την ύπαρξή ασαφών σχέσεων, ασαφών εννοιών, καθώς επίσης και λεκτικά που κατέχουν τον ρόλο των τιμών των ιδιοτήτων, που αποδίδουν και αυτά με την σειρά τους ασαφή γνώση.

Παραδείγματα τέτοιων ασαφών στοιχείων, είναι τα στιγμιότυπα των κλάσεων «**Beginner**» και «**Experienced**», οι ασαφείς σχέσεις «**is_responsible_for**» που συνδέουν τα στιγμιότυπα των κλάσεων «Driver» και «Traffic_Accident» (δηλαδή το ότι ένας οδηγός είναι υπεύθυνος για την πρόκληση ενός ατυχήματος) και τέλος οι τιμές της ιδιότητας «**aged**» (youth, middle-age, old), που κατηγοριοποιούν έναν οδηγό βάση της ηλικίας του.



Εικόνα 8.3- Αρχική σελίδα πλατφόρμας «*wb*GraphFuzzyOnto»



Εικόνα 8.4 - Tab «Step1» επιλογή ασαφών στοιχείων

Εφόσον μας παρέχεται η δυνατότητα μέσω της πλατφόρμας «wbGraphFuzzyOnto», να πραγματοποιηθεί η ασαφοποίηση των στοιχείων μιας οντολογίας και σε μεταγενέστερους περιόδους, επιλέχτηκαν να πραγματοποιηθεί ασαφοποίηση όχι σε όλα τα ασαφή στοιχεία της οντολογίας «Traffic_Accident», αλλά σε αυτά που κρίθηκαν πιο αναγκαία. Συγκεκριμένα επιλέχτηκαν τα εξής στοιχεία:

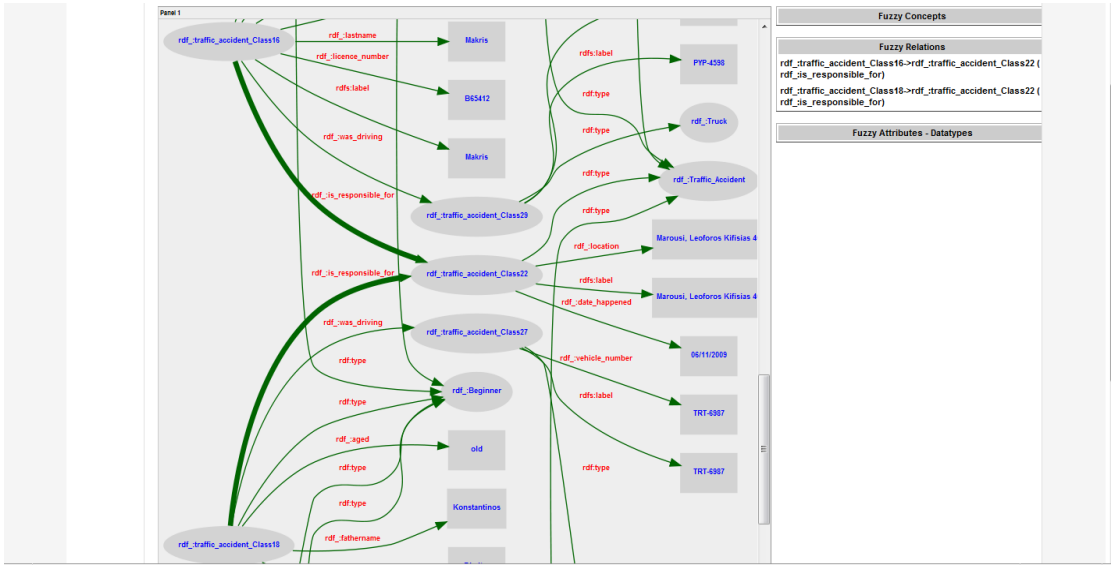
Επιλεγμένες Ασαφείς Σχέσεις:

- **rdf_:is_responsible_for.** Συνδέει το στιγμιότυπο «rdf_:traffic_accident_Class16» της κλάσης «Experienced», το οποίο κατέχει τον ρόλο του υποκείμενου στην RDF πρόταση και αντιστοιχεί στον οδηγό με επίθετο(ιδιότητα rdf_:lastname) την τιμή «Makris», με το στιγμιότυπο «rdf_:traffic_accident_Class22» της κλάσης «Traffic_Accident», το οποίο κατέχει τον ρόλο του αντικείμενου στην RDF πρόταση και αντιστοιχεί στο ατύχημα με σημείο συμβάντος την τιμή «Marousi, Leoforos Kifisias 40».

- **rdf_:is_responsible_for.** Συνδέει το στιγμιότυπο «rdf_:traffic_accident_Class18» της κλάσης «Beginner», το οποίο κατέχει τον ρόλο του υποκείμενου στην RDF πρόταση και αντιστοιχεί στον οδηγό με επίθετο(ιδιότητα rdf_:lastname) την τιμή «Gkini», με το στιγμιότυπο «rdf_:traffic_accident_Class22» της κλάσης «Traffic_Accident», το οποίο κατέχει τον ρόλο του αντικείμενου στην RDF πρόταση και αντιστοιχεί στο ατύχημα με σημείο συμβάντος την τιμή «Marousi, Leoforos Kifisias 40».

Επεξήγηση:

Η σχέση «rdf_:is_responsible_for» σε συνδυασμό με τα στιγμιότυπα που σχετίζετε, αποδίδει ασαφή γνώση, διότι σε ένα τροχαίο ατύχημα ένας οδηγός που συμμετείχε στην πρόκληση του ατυχήματος, μπορεί να είναι υπεύθυνος για το συγκεκριμένο ατύχημα ως προς κάποιο βαθμό. Στην εικόνα 8.5 απεικονίζονται οι επιλεγμένες σχέσεις, οι οποίες αναγράφονται και στην περιοχή «Fuzzy Relations» του tab «Step1».

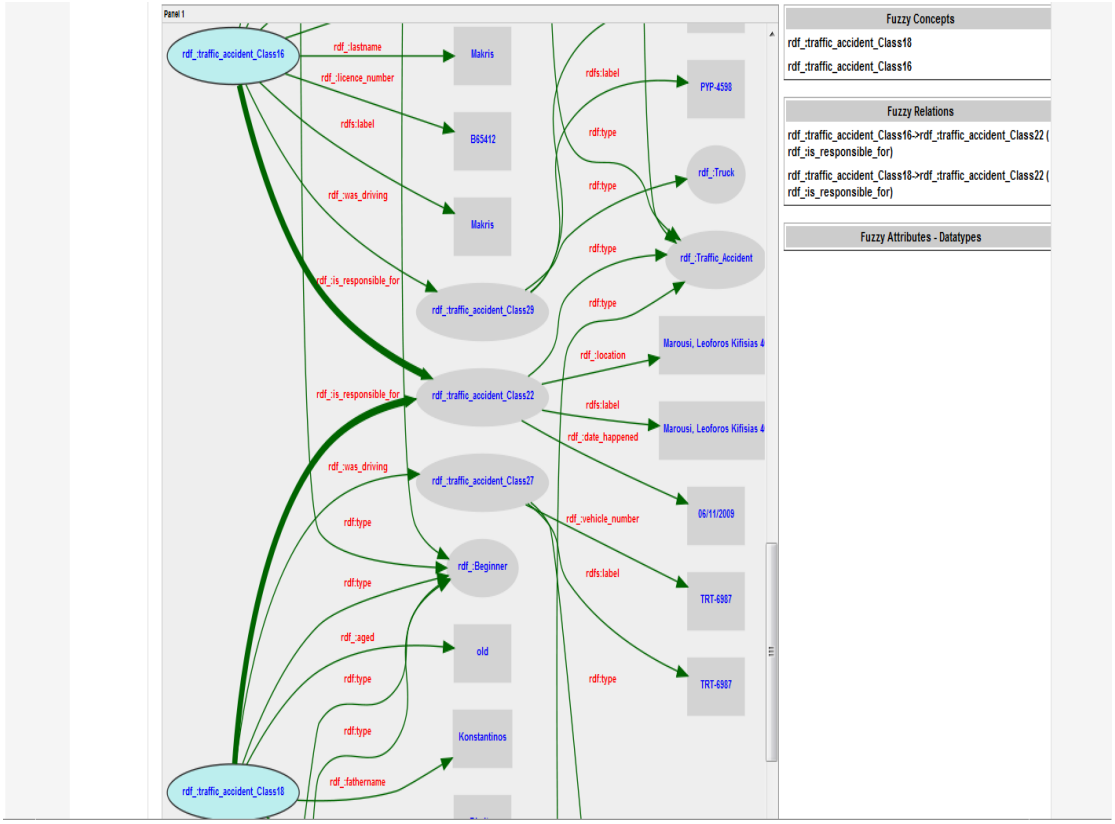


Εικόνα 8.5 – Επιλογή ασαφών σχέσεων

Επιλεγμένες Ασαφείς Έννοιες:

- **rdf_:traffic_accident_Class16.** Στιγμιότυπο της κλάσης «Experienced». Το συγκεκριμένο στιγμιότυπο αποδίδει ασαφή γνώση, διότι δεν είναι ακριβές σε τι βαθμό θεωρείται ένας οδηγός έμπειρος.
- **rdf_:traffic_accident_Class18**. Στιγμιότυπο της κλάσης «Beginner». Το συγκεκριμένο στιγμιότυπο αποδίδει ασαφή γνώση, διότι δεν είναι ακριβές σε τι βαθμό θεωρείται ένας οδηγός αρχάριος.

Στην εικόνα 8.6 απεικονίζονται τα επιλεγμένα στιγμιότυπα των κλάσεων «Experienced» και «Beginner» , τα οποία αναγράφονται και στην περιοχή «Fuzzy Concepts» του tab «Step1».



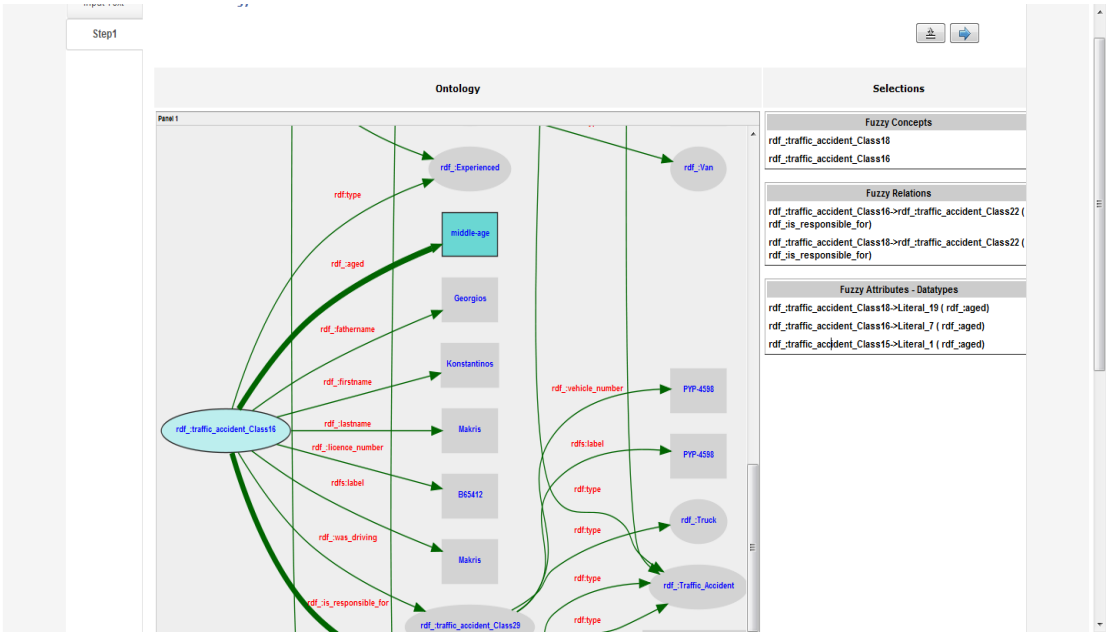
Εικόνα 8.6 – Επιλογή ασαφών εννοιών

Επιλεγμένες Ιδιότητες των οποίων οι τιμές αποδίδουν ασαφή γνώση:

- **rdf_:aged.** Πρόκειται για ιδιότητα του στιγμιότυπου «rdf_:traffic_accident_Class18» της κλάσης «Beginner», στην οποία έχει ορισθεί τιμή το λεκτικό «old».
- **rdf_:aged.** Πρόκειται για ιδιότητα του στιγμιότυπου «rdf_:traffic_accident_Class16» της κλάσης «Experienced», στην οποία έχει ορισθεί τιμή το λεκτικό «middle-age».
- **rdf_:aged.** Πρόκειται για ιδιότητα του στιγμιότυπου «rdf_:traffic_accident_Class15» της κλάσης «Beginner», στην οποία έχει ορισθεί τιμή το λεκτικό «youth».

Επεξήγηση:

Η ιδιότητα «rdf_:aged**»** κατηγοριοποιεί έναν οδηγό βάσης της ηλικίας του. Οι τιμές της ιδιότητας «rdf_:aged**»** αποδίδουν ασαφή γνώση, λόγο της έλλειψης ακριβών ορίων (πεδίο ορισμού) για τις συγκεκριμένες τιμές. Για παράδειγμα, από πια ηλικία μέχρι πια ηλικία θεωρείται ένας οδηγός ηλικιωμένος(old); Στην εικόνα 8.7 απεικονίζεται η επιλεγμένη ιδιότητα «rdf_:aged» και η τιμής της(old), για το στιγμιότυπο «rdf_:traffic_accident_Class18». Οι επιλεγμένες ιδιότητες καθώς επίσης και οι τιμές τους, αναγράφονται και στην περιοχή «Fuzzy Attributes - Datatypes» του tab «Step1».



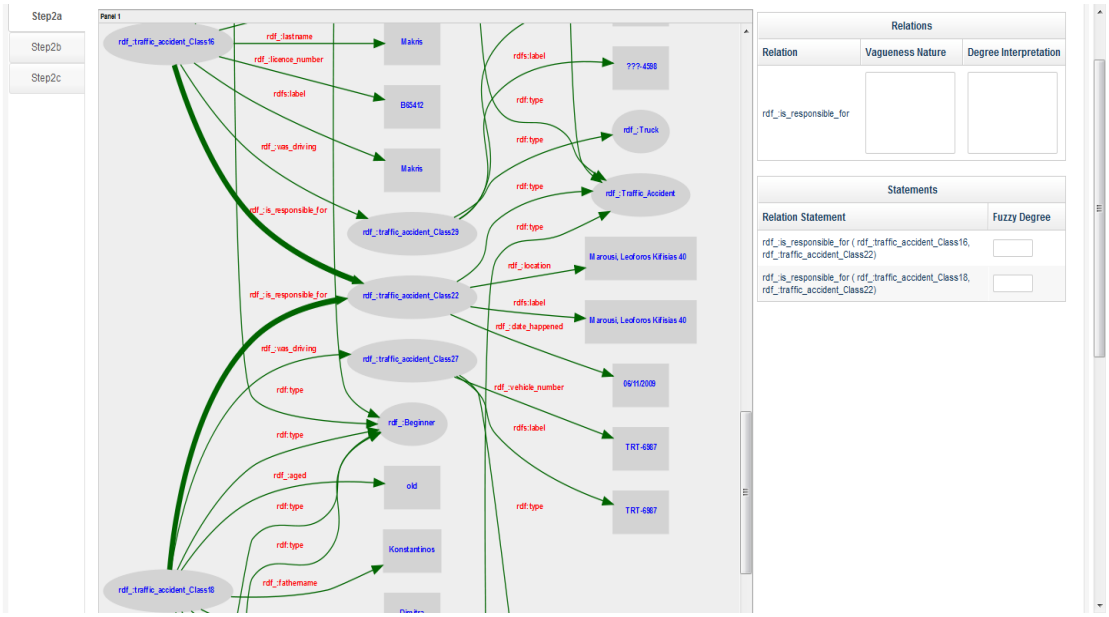
Εικόνα 8.7 - Επιλογή ιδιότητας της οποίας η τιμή αποδίδει ασαφή γνώση

Στο σημείο αυτό, το βήμα 1 που αφορά την επιλογή των ασαφών στοιχείων, έχει ολοκληρωθεί. Πλέον οι μηχανικοί οντολογιών είναι σε θέση να συνεχίσουν την διαδικασία ασαφοποίησης των επιλεγμένων στοιχείων, ανά κατηγορία που ανήκει το κάθε επιλεγμένο στοιχείο.

8.3 Ασαφοποιηση Επιλεγμενων Στοιχειων

Μόλις ο μηχανικός οντολογιών επιλέξει το κουμπί «Next Step»(εικόνα 8.8) από το tab «Step1», μεταφέρεται στο tab «Step2a» (εικόνα 8.9). Το συγκεκριμένο tab, αφορά την περιγραφή των ασαφών σχέσεων, που είχε επιλέξει ο μηχανικός οντολογιών στο προηγούμενο βήμα.

Εικόνα 8.8 – Επιλογή κουμπιού «Next Step» από το tab «Step1»



Εικόνα 8.9 – Tab «Step2a» περιγραφή Ασαφών Σχέσεων

Στον πίνακα «Relations», οι μηχανικοί οντολογιών περιγράφουν τα στοιχεία Vagueness Nature και Degree Interpretation, για κάθε μία από τις επιλεγμένες ασαφείς σχέσεις. Στον πίνακα «Statements», ο ειδικοί της γνωστικής περιοχής προσδιορίζουν τους βαθμούς ασάφειας, για κάθε μία από τις επιλεγμένες σχέσεις, σε συνδυασμό με τα στιγμιότυπα των κλάσεων που αναφέρεται η κάθε μία. Κάνοντας κλικ πάνω σε μία επιλεγμένη σχέση από τον RDF γράφο, επιλέγονται από του πίνακες «Relations» και «Statements», οι έγγραφες που σχετίζονται με την σχέση που επέλεξε ο μηχανικός οντολογιών ή ο ειδικός της γνωστικής περιοχής.

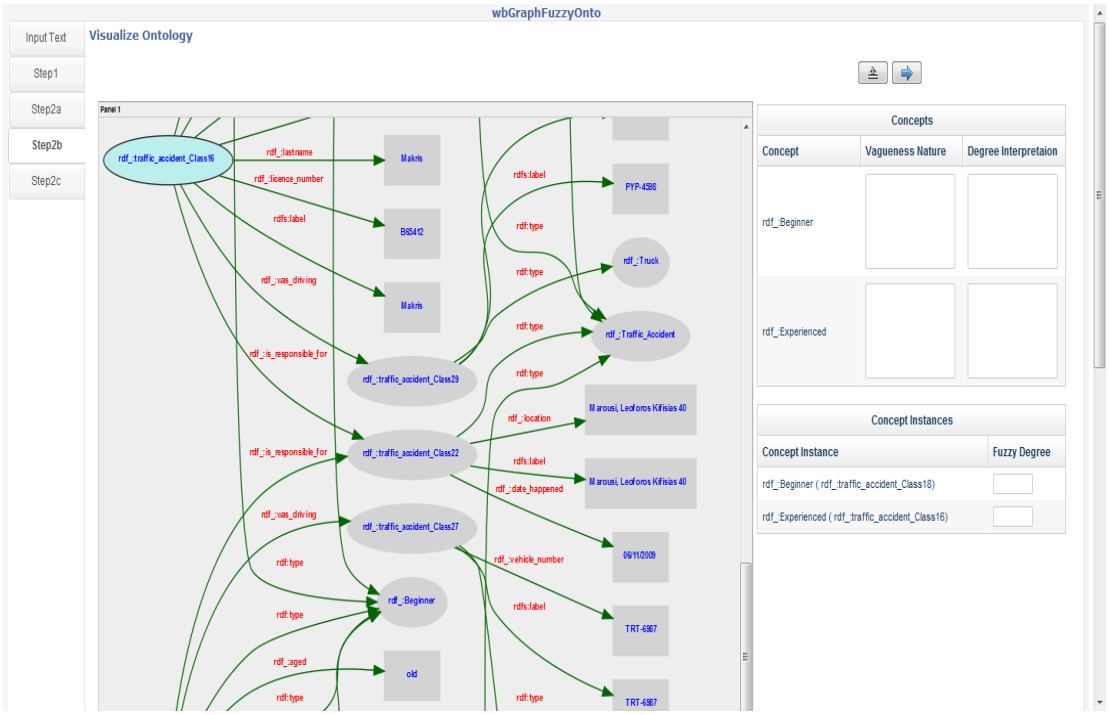
Ξεκινώντας την περιγραφή των σχέσεων, ο μηχανικός προσδιορίζει το Vagueness Nature και Degree Interpretation για την σχέση «**rdf_:is_responsible_for**». Συγκεκριμένα το Vagueness Nature για την σχέση «**rdf_:is_responsible_for**» αντιστοιχεί στο είδος ασάφειας «**Combinatory vagueness**», λόγω της έλλειψης ακριβών στοιχείων που να οριοθετούν έναν οδηγό υπαίτιο, για την πρόκληση ενός τροχαίου ατυχήματος. Το Degree Interpretation για την συγκεκριμένη ασαφή σχέση, έχει να κάνει με τον βαθμό ευθύνης-υπαιτιότητας του οδηγού, για το ατύχημα στο οποίο συμμετείχε στην πρόκληση του.

Στην συνέχεια οι ειδικοί της συγκεκριμένης γνωστική περιοχής, μελετώντας αρχικά τα στοιχεία τα οποία προσδιόρισαν στον πίνακα «Relations» οι μηχανικοί οντολογιών, είναι σε θέση να προσδιορίσουν τους βαθμούς ασάφειας για την σχέση «**rdf_:is_responsible_for**» και τα στιγμιότυπα των κλάσεων στα οποία αναφέρεται κάθε φορά. Στην εικόνα 8.10 απεικονίζονται οι βαθμοί ασάφειας που ορίστηκαν.

Εικόνα 8.10 – Προσδιορισμός βαθμών ασάφειας των Ασαφών σχέσεων σε συνδυασμό με τα στιγμιότυπα στα οποία αναφέρονται

Συγκεκριμένα ορίζεται η τιμή 0,7 που αντιστοιχεί στο στιγμιότυπο «rdf_:traffic_accident_Class16» της κλάσης «Experienced», η οποία αναφέρεται στον οδηγό κ. Μακρή και η τιμή 0,3 που αντιστοιχεί στο στιγμιότυπο «rdf_:traffic_accident_Class18» της κλάσης «Beginner», η οποία αναφέρεται στην οδηγό κα. Γκίνη. Ορίζοντας βαθμό ασάφειας 0,7, δηλώνεται ότι ο κ. Μακρής ήταν υπεύθυνος-υπαίτιος για την πρόκληση του ατυχήματος, που πραγματοποιήθηκε στην Λεωφόρο Κηφισίας στην περιοχή του Μαρουσίου σε βαθμό 0,7 (ή σε ποσοστό 70%), ενώ η κα Γκίνη για το ίδιο ατύχημα, ήτανε υπεύθυνη-υπαίτια σε βαθμό 0,3(ή σε ποσοστό 30%).

Σε αυτό το σημείο η ασαφοποίηση των επιλεγμένων σχέσεων έχει ολοκληρωθεί. Επιλέγοντας ο μηχανικός οντολογιών το κουμπί «Next Step» από το tab «Step2a», μεταφέρεται στο tab «Step2b» (εικόνα 8.11). Το συγκεκριμένο tab, αφορά την περιγραφή των ασαφών εννοιών(κλάσεων) και των στιγμιότυπων τους, που είχε επιλέξει ο μηχανικός οντολογιών σε προηγούμενο βήμα (tab «Step1»).

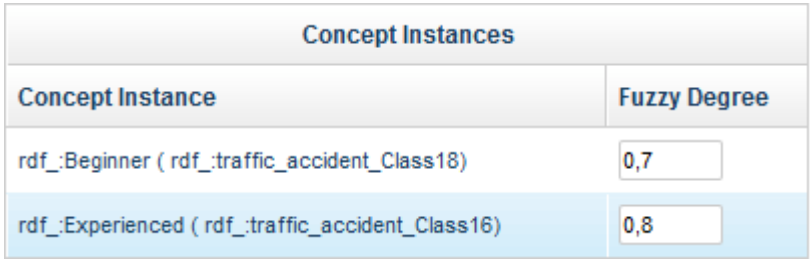


Εικόνα 8.11 - Tab «Step2b» περιγραφή Ασαφών Εννοιών

Στον πίνακα «Concepts», οι μηχανικοί οντολογιών περιγράφουν τα στοιχεία Vagueness Nature και Degree Interpretation, για κάθε μία από τις επιλεγμένες ασαφείς έννοιες. Στον πίνακα «Concept Instances», ο ειδικοί της γνωστικής περιοχής προσδιορίζουν τους βαθμούς ασάφειας, για κάθε ένα από τα επιλεγμένα στιγμιότυπα των ασαφών εννοιών. Δηλαδή, σε τι βαθμό ένα στιγμιότυπο μιας ασαφούς έννοιας θεωρείται στιγμιότυπο της έννοιας αυτής. Κάνοντας κλικ πάνω σε ένα στιγμιότυπο ασαφής έννοιας από τον RDF γράφο, επιλέγονται από του πίνακες «Concepts» και «Concepts Instances», οι έγγραφες που σχετίζονται με το στιγμιότυπο της ασαφούς εννοίας που επέλεξε ο μηχανικός οντολογιών ή ο ειδικός της γνωστικής περιοχής.

Ξεκινώντας την περιγραφή των ασαφών εννοιών, ο μηχανικός προσδιορίζει το Vagueness Nature και Degree Interpretation για τις έννοιες «**rdf_:Beginner**» και «**rdf_:Experienced**». Συγκεκριμένα το Vagueness Nature για την έννοια «**rdf_:Beginner**» αντιστοιχεί στο είδος ασάφειας «**Degree vagueness**», λόγω της έλλειψης ακριβών ορίων, που να περιγράφουν έναν οδηγό αρχάριο. Το Degree Interpretation για την συγκεκριμένη ασαφή έννοια, έχει να κάνει με τον βαθμό στον οποίο θεωρείται ένας οδηγός αρχάριος. Παρομοίως το Vagueness Nature για την έννοια «**rdf_:Experienced**» αντιστοιχεί στο είδος ασάφειας «**Degree vagueness**», λόγω της έλλειψης ακριβών ορίων, που να περιγράφουν έναν οδηγό έμπειρο . Το Degree Interpretation για την έννοια «**rdf_:Experienced**», σχετίζεται με τον βαθμό στον οποίο θεωρείται ένας οδηγός έμπειρος.

Στην συνέχεια οι ειδικοί της συγκεκριμένης γνωστική περιοχής, μελετώντας αρχικά τα στοιχεία τα οποία προσδιόρισαν στον πίνακα «Concepts» οι μηχανικοί οντολογιών, προσδιορίζουν με την σειρά τους, τους βαθμούς ασάφειας για τα στιγμιότυπα «**rdf_:traffic_accident_Class18**» και «**rdf_:traffic_accident_Class16**» των εννοιών(κλάσεων) «**rdf_:Beginner**» και «**rdf_:Experienced**» αντίστοιχα . Στην εικόνα 8.12 απεικονίζονται οι βαθμοί ασάφειας που ορίστηκαν.

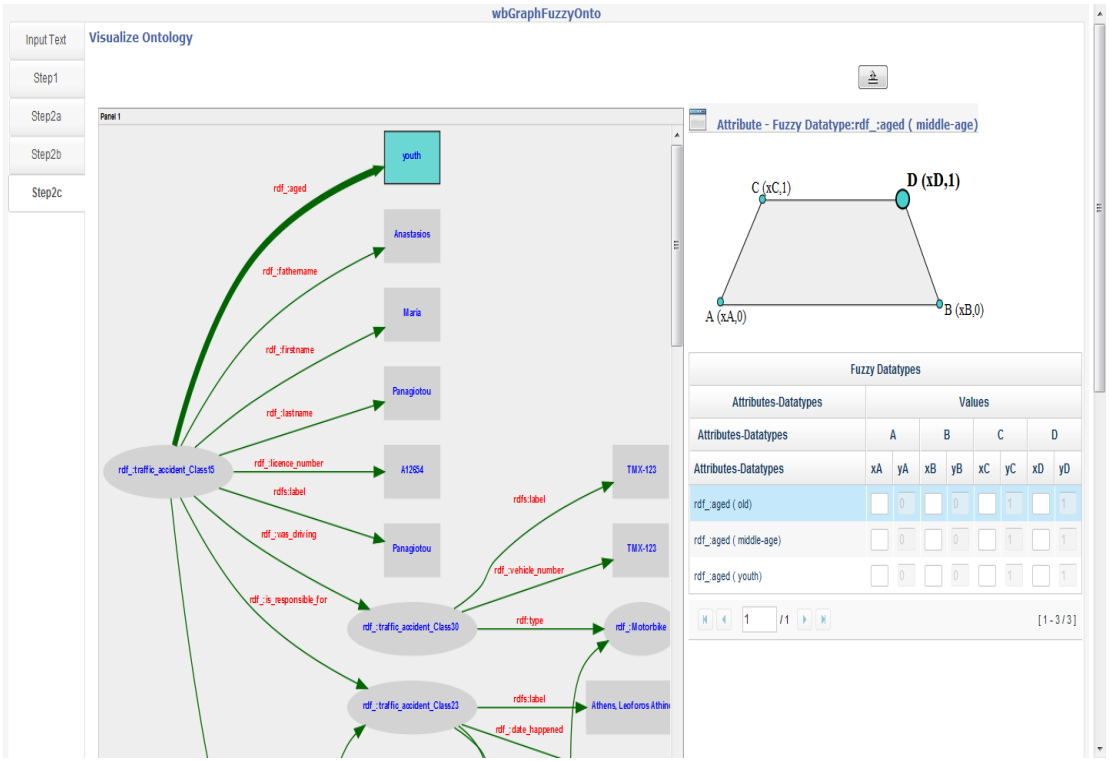


Εικόνα 8.12 - Προσδιορισμός βαθμών ασάφειας στιγμιότυπων Ασαφών Εννοιών

Συγκεκριμένα ορίζεται η τιμή 0,7 που αντιστοιχεί στο στιγμιότυπο «rdf_:traffic_accident_Class18» της κλάσης «Beginner», η οποία αναφέρεται στην οδηγό κα. Γκίνη και η τιμή 0,8 που αντιστοιχεί στο στιγμιότυπο «rdf_:traffic_accident_Class16» της κλάσης «Experienced», η οποία αναφέρεται στον οδηγό κ. Μακρή. Ορίζοντας βαθμό ασάφειας 0,7, δηλώνεται ότι η οδηγός κα. Γκίνη θεωρείται αρχάριος οδηγός, σε βαθμό 0,7(ή σε ποσοστό 70%), ενώ ο κ. Μακρής θεωρείται έμπειρος οδηγός, σε βαθμό 0,8 (ή σε ποσοστό 80%).

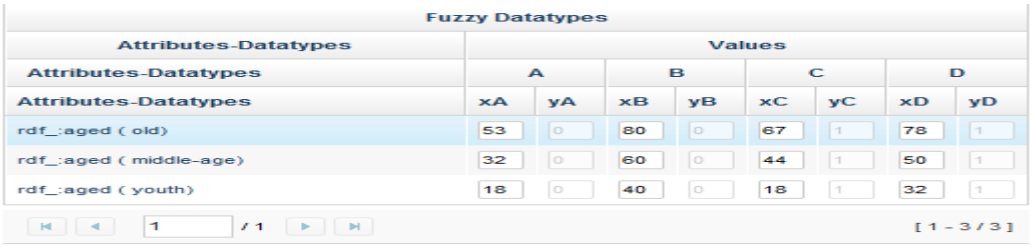
Με τον προσδιορισμό και των βαθμών ασάφειας των στιγμιότυπων των ασαφών εννοιών, η διαδικασία ασαφοποίησης των επιλεγμένων εννοιών έχει ολοκληρωθεί.

Επιλέγοντας ο μηχανικός οντολογιών το κουμπί «Next Step» από το tab «Step2b», μεταφέρεται στο tab «Step2c» (εικόνα 8.13). Στο συγκεκριμένο tab, πραγματοποιείται μέσω της πλατφόρμας «wbGraphFuzzyOnto», η ασαφοποίηση των ασαφών λεκτικών ορών που αποτελούν τιμή σε μία επιλεγμένη ιδιότητα.



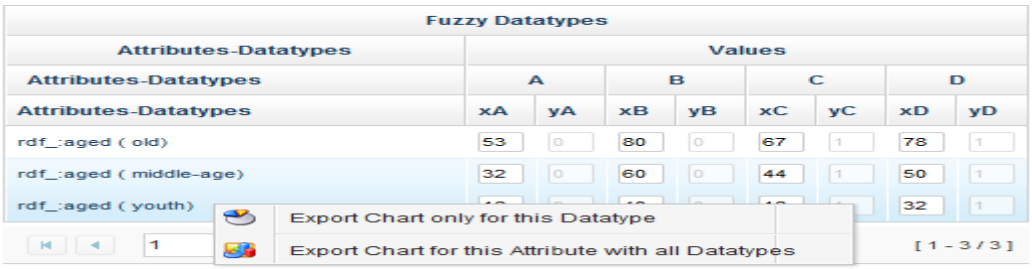
Εικόνα 8.13 - Tab «Step2c» περιγραφή τιμών των ιδιοτήτων που αποδίδουν ασαφή γνώση

Συγκεκριμένα, οι ειδικοί της γνωστικής περιοχής προσδιορίζουν το σύνολο τιμών του κάθε ασαφή λεκτικού όρου, που αποτελεί τιμή σε μία επιλεγμένη ιδιότητα. Στόχος του κάθε μέλους της ομάδας των ειδικών, είναι η δημιουργία ενός γραφικού διαγράμματος εφάμιλλο με το διάγραμμα της γλωσσικής μεταβλητής «Απόδοσης Μηχανής» που απεικονίζεται στην εικόνα 4.2, από το οποίο προκύπτει και ο βαθμός συμμετοχής μιας τιμής χ σε ένα ασαφές σύνολο, όπου το συγκεκριμένο σύνολο αντιστοιχεί σε έναν λεκτικό όρο. Το επιτραπέζιο σχήμα το οποίο περικλείεται στο tab «Step2c», συμβάλει προς αυτήν την κατεύθυνση. Τοποθετώντας τον δείκτη του ποντικού, σε ένα από τα τέσσερα σημεία του επιτραπέζιου σχήματος κάθε φορά, εμφανίζεται κατάλληλο μήνυμα το οποίο ενημερώνει τον χρήστη για τον σωστό τρόπο συμπλήρωσης των πεδίων του πίνακα «Fuzzy Datatypes». Στην εικόνα 8.14 απεικονίζονται τα σύνολα τιμών που οριστήκαν για τους ασαφής λεκτικούς ορούς «**old**», «**middle**-age» και «**youth»** της ιδιότητας «**rdf_:aged**».



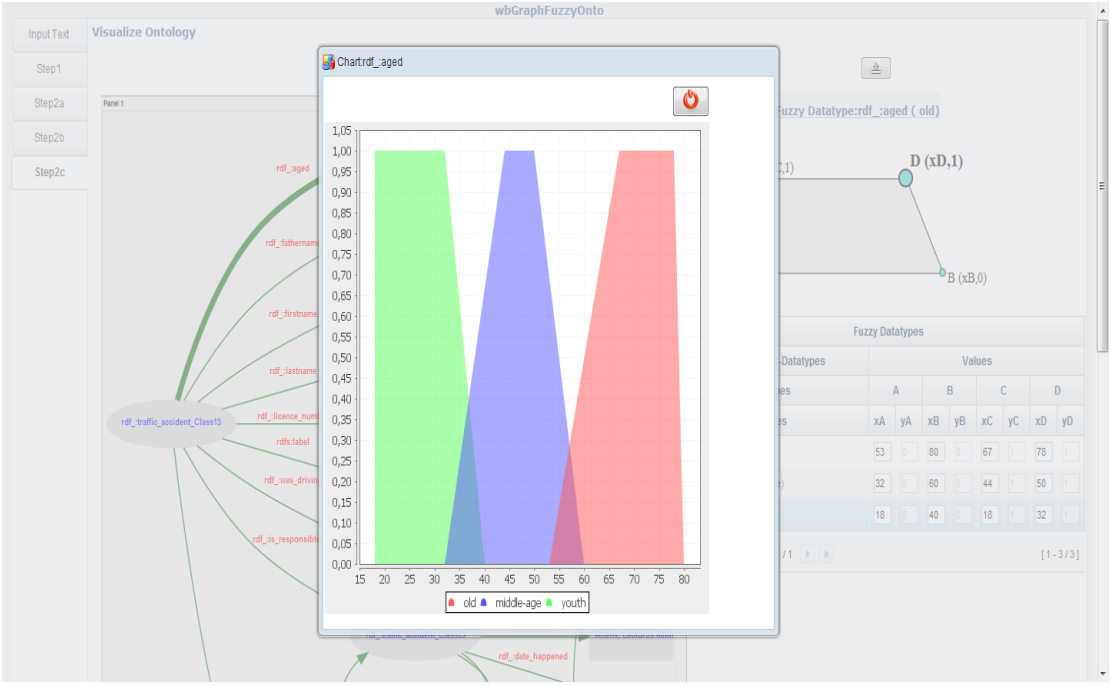
Εικόνα 8.14 – Περιγραφή συνόλου τιμών Ασαφών Λεκτικών Όρων

Κάνοντας αριστερό κλικ πάνω σε μία εγγραφή του πίνακα «Fuzzy Datatypes», εμφανίζεται ένα μενού επιλογών (εικόνα 8.15), μέσω του οποίου δίνεται η ευχέρεια στον χρήστη να εμφανίσει ένα γραφικό διάγραμμα, που απεικονίζει το σύνολο τιμών που ορίστηκε για το επιλεγμένο ασαφή λεκτικό όρο ή να εμφανίσει ένα γραφικό διάγραμμα, που απεικονίζει το σύνολο τιμών που αφορούν τις τιμές μιας ιδιότητας (εφάμιλλο της διαγράμματος της εικόνας 4.2).



Εικόνα 8.15 – Μενού Επίλογων παραγωγής γραφικών διαγραμμάτων

Επιλέγοντας την δεύτερη επιλογή, εμφανίζεται το γραφικό διάγραμμα που απεικονίζεται στην εικόνα 8.16 και που αφόρα την ιδιότητα«**rdf_:aged**».



Εικόνα 8.16 – Διάγραμμα αναπαράστασης Ασαφών Γλωσσικών Όρων

Από το συγκεκριμένο γραφικό διάγραμμα μπορούμε να συμπεραίνουμε, ότι ένας οδηγός με ηλικία 35 χρονών, θεωρείται μεσήλικας(middle-age) με βαθμό 0,26 και νέος(youth) σε βαθμό 0,55.

Μετά τον προσδιορισμό του συνόλου τιμών, του κάθε ασαφή λεκτικού όρου, που αποτελεί τιμή σε μία επιλεγμένη ιδιότητα, από τους ειδικούς της γνωστικής περιοχής, η ασαφοποίηση των λεκτικών ορών της κάθε επιλεγμένης ιδιότητας έχει ολοκληρωθεί. Με το πέρας και του συγκεκριμένου βήματος, έχει επιτευχτεί μέσω της πλατφόρμας

«wbGraphFuzzyOnto», η ασαφοποίηση όλων των επιλεγμένων στοιχείων. Οι μηχανικοί οντολογιών επιλέγοντας το κουμπί «Export to XML» , που βρίσκεται στο tab «Step2c», παράγουν ένα XML αρχείο, με τα στοιχεία της RDF σημασιολογικής γλώσσας που συνθέτουν την οντολογία «Traffic_Accident», καθώς επίσης και όλων των επίλογων που πραγματοποίησαν, των στοιχείων που όρισαν , καθόλη την διάρκεια της ασαφοποίησης της οντολογίας «Traffic_Accident», μέσω της πλατφόρμας «wbGraphFuzzyOnto». Το συγκεκριμένο XML έγγραφο, μπορεί να χρησιμοποιηθεί στο μέλλον, από τους μηχανικούς οντολογιών και τους ειδικούς της γνωστικής περιοχής, για την περιγραφή και άλλων στοιχείων που αποδίδουν ασαφή γνώση, της οντολογίας «Traffic_Accident».

9 ΣΥΜΠΕΡΑΣΜΑΤΑ – ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ

Το κεφάλαιο αυτό εστιάζεται στα συμπεράσματα στα οποία οδηγεί η εκπόνηση αυτής της διπλωματικής εργασίας και στις μελλοντικές επεκτάσεις που ενδεχομένως μπορούν να γίνουν.

9.1 ΣΥΜΠΕΡΑΣΜΑΤΑ

Ο Σημασιολογικός Ιστός είναι η μία επέκταση του «Παγκόσμιου Ιστού» που επιτρέπει την επεξεργασία και διαχείριση του περιεχομένου του ιστού σε επίπεδο μηχανών δηλαδή από τους υπολογιστές, συμβάλλοντας έτσι στην μείωση πολλών μειονεκτημάτων που παρουσιάζουν οι υπηρεσίες που παρέχονται από τον σημερινό παγκόσμιο ιστό.

Στην παρούσα διπλωματική εργασία πραγματοποιήθηκε μία ποιοτική περιγραφή του όρου Σημασιολογικού Ιστού και κυρίως του κύριου χαρακτηριστικού που βασίζεται ο Σημασιολογικός Ιστός, μέσω του οποίου επιτυγχάνεται η αναπαράσταση γνώσης και ο συμπερασμός, που είναι οι «Οντολογίες». Πραγματοποιήθηκε λοιπόν μια λεπτομερής περιγραφή των χαρακτηριστικών και της δομής μιας οντολογίας, των κυριότερων σημασιολογικών γλωσσών περιγραφής οντολογιών καθώς και μεθοδολογιών περιγραφής και ανάπτυξης οντολογιών. Εστιάσαμε περισσότερο στην περιγραφής ενός προβλήματος που συναντάμε κατά την δημιουργία μιας οντολογίας που έχει να κάνει με την αναπαράσταση της ασαφούς γνώσης δηλαδή της δυσκολία που συναντάμε στο να περιγράψουμε με μαθηματική ακρίβεια τον κόσμο που μας περιβάλει.

Στα πλαίσια του προβλήματος αυτού και της εξάλειψης του, αναφερθήκαμε στον ορό της «Ασάφειας» και στην δημιουργία «Ασαφών Οντολογιών». Παρουσιάστηκε λεπτομερέστατα μία μεθοδολογία ανάπτυξης Ασαφών Οντολογιών η «IKARUS-Onto». Βασιζόμενοι κυρίως στην συγκεκριμένη μεθοδολογία αναπτύξαμε μια διαδικτυακή πλατφόρμα ασαφοποίησης οντολογιών την «*wb*GraphFuzzyOnto». Η συγκεκριμένη πλατφόρμα σχεδιάστηκε και υλοποιήθηκε βάση τεχνολογιών που ανήκουν σε μία νέα εξελικτική τάση που επικρατεί στον χώρο του παγκοσμίου ιστού και είναι το Web 2.0.

Με την υλοποίηση της πλατφόρμας «*wb*GraphFuzzyOnto» και από τα αποτελέσματα χρήσης της αναδεικνύεται η αναγκαιότητα για την ύπαρξη συστημάτων ασαφοποίησης οντολογιών, επιτυγχάνοντας με αυτόν τον τρόπο την αποδοτικότερη και εγκυρότερη διαχείριση της πληροφορίας του ιστού σε επίπεδο μηχανών και προς όφελος ως προς την εξυπηρέτηση των αναγκών, των χρηστών του παγκοσμίου ιστού.

9.2 ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ – ΑΝΟΙΚΤΑ ΘΕΜΑΤΑ

Στα πλαίσια της εκπόνησης της παρούσας διπλωματικής εργασίας καταγράφηκαν μερικές ιδέες για μελλοντικές επεκτάσεις της πλατφόρμας «*wb*GraphFuzzyOnto». Συγκεκριμένα αναφέρονται:

- Δημιουργία ενός πιο ολοκληρωμένου συστήματος με δυνατότητες ορισμού συγκεκριμένων δικαιωμάτων σε κάθε ομάδα χρηστών.
- Χρησιμοποίηση βάσης δεδομένων για την αποθήκευση συμβατικών και ασαφών οντολογιών.
- Εφόσον έχει πραγματοποιηθεί η ασαφοποίηση σε μία οντολογία, επιλογή και διατύπωση των ασαφών ορών με χρήση μιας σημασιολογικής γλώσσας περιγραφής ασάφειας, όπως για παράδειγμα η f-OWL.
- Δυνατότητα αναζήτησης και ανάκτησης υλοποιημένων οντολογιών βάση συγκεκριμένων στοιχείων , όπως για παράδειγμα αν πρόκειται για συμβατική ή ασαφής οντολογία, αν έχει διατυπωθεί σε μια συγκεκριμένη σημασιολογική γλώσσα ή αν έχει πραγματοποιηθεί ασαφοποίηση σε μία οντολογία ως προς κάποιο βήμα.
- Δυνατότητα ελέγχου μίας υλοποιημένης ασαφής οντολογίας ως προς τις αρχές που θα πρέπει να την διέπουν και να την χαρακτηρίζουν έγκυρη. Τέτοιες αρχές περιγράφτηκαν στο κεφάλαιο 4.3.4.

10 Βιβλιογραφια

[1]. http://en.wikipedia.org/wiki/Web_2.0

[2]. **Deursen, Ali Mesbah & Arie van.** An Architectural Style for Ajax. s.l. *: TUDelft-SERG, 2006.*

[3]. **Paulson, Linda Dailey.** Building Rich Web Applications with Ajax. *IEEE ComputerMagazine.* October 2005.

[4]. http://www.w3.org/TR/xmlschema-2/

[5]. http://www.w3.org/RDF/

[6]. **T. Berners-Lee, J. Hendler, and O. Lassila**. The Semantic Web - A New Form of Web Content that is Meaningful to Computers Will Unleash a Revolution of New Possibilities. Scientific American, 284(5):34–43, 2001.

[7]. **Grigoris Antoniou & Frank van Harmelen.** A Semantic Web Primer, Second Edition, The MIT Press, Cambridge, Massachusetts, London, England, 2004.

[8]. **Franz Baader, Ian Horrocks and Ulrike Sattler, In Frank van Harmelen, Vladimir Lifschitz and Bruce Porter, editors.** Handbook of Knowledge Representation, Elsevier 2007.

[9]. http://techwiki.openstructs.org/index.php/Metamodeling_in_Domain_Ontologies

[10]. **L. A. Zadeh**. "Fuzzy sets", Department of Electrical Engineering and Electronics Research Laboratory, University of California, Berkeley, California, 1965.

[11]. **G. Stoilos, G. Stamou, V. Tzouvaras, J.Z. Pan, and I. Horrocks**. Fuzzy OWL: Uncertainty and the semantic web. In *Proc. of the International Workshop on OWL: Experiences and Directions,* 2005.

[12]. **Jeff Z. Pan, Giorgos Stamou, Giorgos Stoilos, Stuart Taylor, Edward Thomas.** Scalable Querying Services over Fuzzy Ontologies, Beijing, China, 2008**.**

[13]. http://www-ksl.stanford.edu/kst/what-is-anontology.html

[14]. http://www.w3.org/TR/webont-req/#section-introduction

[15]. http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html

[16]. http://tomgruber.org/writing/onto-design.htm

[17]. http://starlab.vub.ac.be/teaching/uschold.pdf

[18]. **Mike Uschold, Martin King.** Towards a Methodology for Building Ontologies, IJCAI-95, July 1995.

[19]. **York Sure, Steffen Staab, Rudi Studer**. Methodology for Development and Employment of Ontology based Knowledge Management Applications, IST-1999-10132 On-To-Knowledge, December 2002.

[20]. **Mariano Fernandez, Asuncion Gomez-Perez, Natalia Juristo.** METHONTOLOGY: From Ontological Art Towards Ontological Engineering, Madrid, Spain.

[21]. **H.Sofia Pinto, Steffen Staab, Christoph Tempich.** DILIGENT: Towards a fine-grained methodology for Distributed, Loosely-controlled and evolvInG Engineering of oNTologies, European Conference on Artficial Intelligence, Spain, 2004.

[22]. **Panos Alexopoulos, Manolis Wallace, Konstantinos Kafentzis, Dimitris Askounis.** A Methodology for Developing Fuzzy Ontologies, ESCW, Heraklion, Greece, 2010.

[23]. http://pelopas.uop.gr/~gouscos/reports/SocialSoftwareWeb20.pdf

[24]. http://www.observe.gr/files/meletes/WEB%202.0.pdf

[25]. http://oreilly.com/web2/archive/what-is-web-20.html

[26]. http://download.oracle.com/javaee/6/tutorial/doc/

[27]. http://www.w3.org/TR/xhtml1/

[28]. http://www.w3.org/Graphics/SVG/

[29]. http://en.wikipedia.org/wiki/Scalable_Vector_Graphics

[30]. **Marty Hall, Larry Brown.** Core Servlets and JavaServer Pages**:** Volume I: Core Technologies, 2nd Edition, Prentice Hall PTR, 2003.

[31]. http://en.wikipedia.org/wiki/Css

[32]. **J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K.Wilkinson.** Jena: Implementing the Semantic Web Recommendations. Technical Report HPL-2003-146, Hewlett-Packard, December 2003.

[33]. http://www.jfree.org/jfreechart/

[34]. **Emden Gansner and Eleftherios Koutsofios and Stephen North**. Drawing graphs with dot, AT&T Bell Laboratories, January 2006.

[35]. http://books.zkoss.org/wiki/ZK_Developer%27s_Reference

[36]. http://en.wikipedia.org/wiki/Rich_Internet_application

[37]. http://en.wikipedia.org/wiki/Document_Object_Model

[38]. Http://www.w3.org/RDF/Validator/

[39]. http://smi-protege.stanford.edu/repos/protege/protege-core/trunk/

11 ΠΙΝΑΚΑΣ ΑΚΡΩΝΥΜΙΑ

AJAX	Asynchronous Javascript And Xml
API	Application Programing Interface
CSS	Cascade Style Sheet
GIF	Graphics Interchange Format
HTML	HyperText Markup Language
J2EE	Java 2 platform, Enterprise Edition
OWL	Web Ontology Language
PNG	Portable Network Graphics
RDF	Resource Description Framework
RDFS	RDF Schema
SVG	Scalable Vector Graphics
SW	Semantic Web
W3C	World Wide Web Consortium
wbGraphFuzzyOnto	Web-BasedGraphFuzzyOnto
WWW	World Wide Web
XHTML	eXtensible HyperText Markup Language
XML	eXtensible Markup Language

12 Παραρτημα Α – Εγκαταστση Συστηματος

Για να πραγματοποιήσουμε την εγκατάσταση του συστήματος στον εξυπηρετητή (server), χρειαζόμαστε να εγκαταστήσουμε ένα web server ο οποίος να υποστηρίζει εφαρμογές οι οποίες είναι υλοποιημένες σε Java και να εγκαταστήσουμε το λογισμικό GraphViz 2.26.3.

12.1 Εγκαταστση Web Server

Ο Web Server που χρησιμοποιηθούμε είναι ο Apache Tomcat 6.0.26 . Μετάβαση στην τοποθεσία «tomcat.apache.org» από την οποία μπορεί να πραγματοποιηθεί η λήψη της έκδοσης 6.0.26.

Θα πρέπει να επισημάνουμε ότι ο συγκεκριμένος Web Server χρησιμοποίει την θύρα (port) 8080. Σε περίπτωση που χρησιμοποιείται η συγκεκριμένη θύρα από κάποια άλλη διεργασία στον εξυπηρετητή, θα πρέπει να τροποποιηθεί το αρχείο server.xml (τοποθεσία αρχείου ../apache-tomcat-6.0.26/conf) και να ορίσουμε κάποια άλλη τιμή που δεν χρησιμοποιείται από κάποια διεργασία (εικόνα 12.1).

```
<!-- A "Connector" represents an endpoint by which requests are received
     and responses are returned. Documentation at :
     Java HTTP Connector: /docs/config/http.html (blocking & non-blocking)
     Java AJP  Connector: /docs/config/ajp.html
     APR (HTTP/AJP) Connector: /docs/apr.html
     Define a non-SSL HTTP/1.1 Connector on port 8080
-->
<Connector port="8081" protocol="HTTP/1.1"
           connectionTimeout="20000"
           redirectPort="8443" />
```

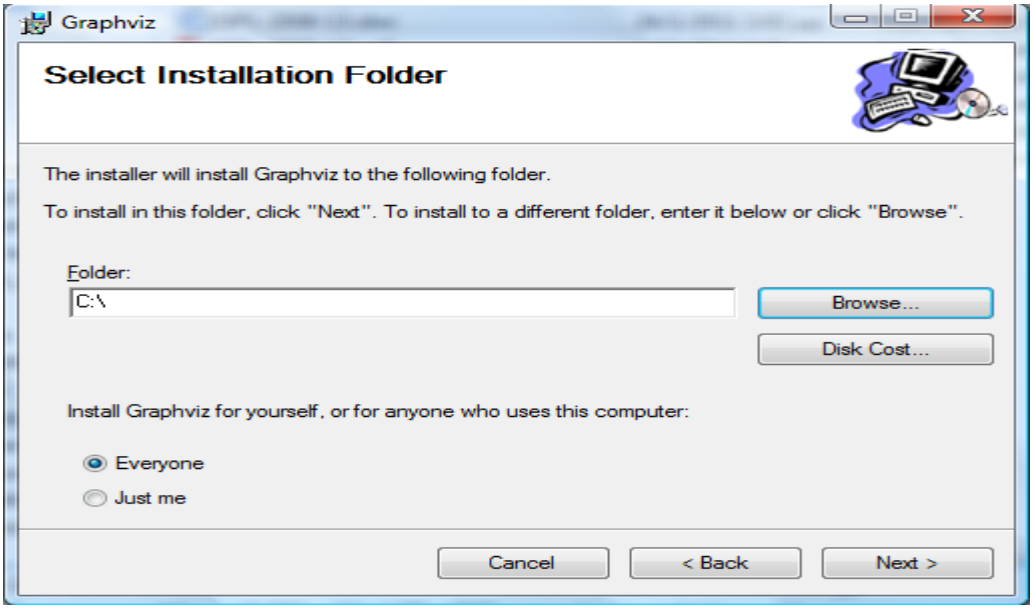
Εικόνα 12.1– Αρχείο Server.xml

12.2 Εγκαταστση GraphViz 2.26.3

Ένας σημαντικός παράγοντας για την σωστή, έγκυρη και ομαλή λειτουργία του συστήματος είναι η εγκατάσταση του Graph Visualization. Πρόκειται για ένα λογισμικό open-source Μετάβαση στην τοποθεσία «http://www.graphviz.org/Download_windows.php» από την οποία μπορεί να πραγματοποιηθεί η λήψη της έκδοσης 2.26.3.

Το Graph Visualization είναι απαραίτητο για την αναπαράσταση των οντολογιών σε γράφους χρησιμοποιώντας σαν είσοδο ένα αρχείο τύπου DOT.

Κατά την διαδικασία εγκατάστασης απαιτείται να ορισθεί η τοποθεσία εγκατάστασης (εικόνα 12.2). Η τοποθεσία η οποία πρέπει να οριστεί έχει την διαδρομή C:\Graphviz2.26.3 .



Εικόνα 12.2- Εγκατάσταση GraphViz 2.26.3

Μόλις ολοκληρωθεί η εγκατάσταση πρέπει να δημιουργηθεί ένα αρχείο τύπου folder με την ονομασία «tmp» μέσα στο φάκελο του Graph Visualization (τοποθεσία C:\Graphviz2.26.3\ ).

12.3 Επαληθευση Έγκυρης Εγκαταστασης

Στο φάκελο εγκατάστασης του Apache Tomcat υπάρχει ένας φάκελος με όνομα webapps. Εκεί αντιγράφουμε το αρχείο FuzzyOnto.war το οποίο μόλις ο server εκκινήσει (εκτελώντας το αρχείο run.bat βρίσκεται στον φάκελο bin του Apache Tomcat) θα επεκταθεί στην αντίστοιχη web εφαρμογή και θα είναι προσβάσιμο από έναν web browser στη διεύθυνση http://localhost:8080/ σε συνδυασμό πάντα και με την θύρα που έχουμε ορίσει. Αν εμφανιστεί η σελίδα με επιτυχία σημαίνει ότι όλα έχουν εγκατασταθεί σωστά.

12.4 Απαιτησεις σε Λογισμικο

Το μηχάνημα στο οποίο τρέχει ο κεντρικός εξυπηρετητής δεν χρειάζεται να τρέχει κάποιο συγκεκριμένο λειτουργικό σύστημα καθώς η τεχνολογία Java που χρησιμοποιήθηκε σε όλα τα υποσυστήματα είναι διαθέσιμη για πάνω από 20 διαφορετικά λειτουργικά συστήματα. Η εγκατάσταση του Apache Tomcat και του Graph Visualization που έγινε αναφορά παραπάνω έχει πραγματοποιηθεί σε περιβάλλον Microsoft Windows Vista Home Premium.